



WebWorks ePublisher Platform

Designer's Guide

Version 2018.1



Table of Contents

WebWorks ePublisher Documentation.....	
WebWorks ePublisher Documentation.....	1
Contacting Quadralay.....	2
Conventions.....	3
Introduction to the WebWorks ePublisher Platform.....	5
What Is ePublisher?.....	6
Workflow.....	7
WebWorks ePublisher Platform Components.....	8
Supported Input Formats.....	9
Supported Output Formats.....	10
How ePublisher Helps You.....	11
Streamline and Automate the Content Publishing Process.....	12
Produce High Quality Deliverables with Fewer Individual Dependencies.....	13
Reduce Support Costs and Increase Customer Satisfaction.....	14
Quickly Update and Deliver Content More Often.....	15
Reduce Content Management Life Cycle Costs.....	16
How Organizations Use ePublisher.....	17
Automatically Update Content on Web Sites and Wikis.....	18
Deliver Full-Featured, Context-Sensitive Help Systems.....	19
Produce Single-Sourced Print and Online Optimized Content.....	20
Features Available in Each Output Format.....	21
Planning and Installing ePublisher.....	24
Licensing Considerations.....	25
Components and Supported Configurations.....	26
Requirements.....	27
ePublisher Express, ePublisher Designer, and ePublisher AutoMap Requirements.....	28
Additional Input Format Requirements.....	29
Additional Output Format Requirements.....	30
Dynamic HTML.....	31
eBook - ePUB 2.0.....	32
Eclipse Help.....	33
Microsoft HTML Help 1.x.....	34
Oracle Help.....	35

PDF.....	36
PDF -XSL-FO.....	37
Sun JavaHelp 2.0.....	38
WebWorks Help 5.0.....	39
WebWorks Reverb.....	40
Configuring web server for Reverb.....	41
Reverb browser requirements.....	42
Wiki - Confluence.....	43
Wiki - MediaWiki.....	44
Wiki - MoinMoin.....	45
Downloading ePublisher Installers.....	46
Microsoft Windows Requirements.....	47
Downloading and Installing the Microsoft .NET 4.6.1 Framework.....	48
Enabling JavaScript.....	49
Installing ePublisher.....	50
Installation Order for ePublisher Components.....	51
Installing ePublisher Components.....	52
Configuring AutoMap for Microsoft Word Inputs.....	54
Understanding Installed Sample Projects and Stationery.....	56
Working with Contract IDs.....	57
Viewing Licensing and Contract ID Information.....	58
Obtaining Contract IDs.....	59
Entering Contract IDs.....	60
Managing Licensing in Environments without Internet Connectivity.....	61
Updating Licensing.....	62
Deactivating Licensing.....	63
Upgrading from Previous Versions.....	64
Updating ePublisher installation.....	65
Preparing existing projects for ePublisher Upgrade.....	66
Upgrading Typical ePublisher Implementations.....	67
Upgrading Implementations with Advanced Customizations.....	68
Upgrading Advanced Customizations of WebWorks Reverb Skin.....	69
Uninstalling ePublisher.....	71
Troubleshooting Installation, License Keys, and Uninstallation.....	72
Problems Installing ePublisher.....	73
Error: Microsoft .NET Framework 2.0 Not Installed.....	74
Error: Please Close all Running Sessions of Microsoft Word.....	75

Problems with Contract IDs and Licensing.....	76
No Contract ID Received.....	77
Error: No Valid License Key Found.....	78
Other Contract ID and Licensing Problems.....	79
Problems Uninstalling ePublisher.....	80
Error: You Must Remove the Previous Version of ePublisher.....	81
Problems Completely Uninstalling ePublisher.....	82
Other Errors Uninstalling ePublisher.....	83
Selecting Input and Output Formats.....	85
Understanding the ePublisher Process and Workflow.....	86
Selecting Your Input Formats.....	87
Selecting Your Output Formats.....	88
Understanding Input Formats.....	89
Adobe FrameMaker.....	90
Microsoft Word.....	91
DITA XML Files.....	92
WebWorks Markdown.....	93
Understanding Output Formats.....	94
Dynamic HTML.....	95
Dynamic HTML Output Viewer.....	97
Delivering Dynamic HTML.....	98
eBook - ePUB 2.0.....	99
Eclipse Help.....	100
Eclipse Help Viewer.....	101
Delivering Eclipse Help.....	102
Microsoft HTML Help.....	103
Benefits of Microsoft HTML Help.....	104
Restrictions and Requirements for Microsoft HTML Help.....	105
HTML Help Viewer.....	106
Toolbar Pane in HTML Help.....	107
Navigation Pane in HTML Help.....	108
Topic Pane in HTML Help.....	109
Topic Only View in HTML Help.....	110
Understanding HTML Help Workshop.....	111
HTML Help Project File (.hhp).....	112
HTML Help Contents File (.hhc).....	113

HTML Help Index File (.hhk).....	114
HTML Help Mapping File (.h).....	115
Delivering HTML Help.....	116
Oracle Help.....	117
Oracle Help Viewer.....	118
Understanding Oracle Help Files.....	119
Helpset .hs File in Oracle Help.....	120
Control .xml Files.....	121
Full Text Search .idx Index File.....	122
Manifest .mft File.....	123
Delivering Oracle Help.....	124
PDF.....	125
PDF - XSL-FO.....	126
Sun JavaHelp.....	127
Sun JavaHelp Viewer.....	128
Understanding Sun JavaHelp Files.....	129
Helpset .hs File in Sun JavaHelp.....	130
Contents toc.xml File in Sun JavaHelp.....	132
Index ix.xml File in Sun JavaHelp.....	133
Map .jhm File in Sun JavaHelp.....	134
Delivering Sun JavaHelp.....	135
WebWorks Help.....	136
The Frameset View in WebWorks Help.....	137
Navigation Pane in WebWorks Help.....	138
Toolbar Pane in WebWorks Help.....	139
Topic Pane in WebWorks Help.....	140
Topic Only View in WebWorks Help.....	141
WebWorks Help Output Files.....	142
Delivering WebWorks Help.....	143
Searching WebWorks Help.....	144
WebWorks Reverb.....	145
Choosing a Skin in WebWorks Reverb.....	146
Choosing the Compact version of a WebWorks Reverb Skin.....	150
Complete List of WebWorks Reverb Skins.....	151
Previewing WebWorks Reverb.....	152
Delivering WebWorks Reverb.....	153
Searching WebWorks Reverb - Microsoft IIS Search.....	154

Searching WebWorks Reverb - Client-side Search.....	155
Searching WebWorks Reverb - URL Method.....	156
Indexing WebWorks Reverb - Client-Side Search for Source Documents, Baggage Files and External URLs.....	157
Using Tidy for Indexing HTML Pages.....	158
Assigning Relevance Weight to Your Source Documents Styles.....	159
Assigning Relevance Weight to Your HTML and PDF Baggage Files.....	160
Search Highlighting in Baggage Files - Client-Side Search.....	162
URL Toggle in WebWorks Reverb.....	163
End-user requirements in WebWorks Reverb.....	164
WebWorks Reverb 2.0.....	165
Choosing a Skin in WebWorks Reverb 2.0.....	166
Using SASS To Customize The WebWorks Reverb 2.0 Interface Design.....	171
Previewing WebWorks Reverb 2.0.....	172
Delivering WebWorks Reverb 2.0.....	173
Understanding Top-Level Groups in WebWorks Reverb 2.0.....	174
Searching WebWorks Reverb 2.0- Client-side Search.....	175
Searching WebWorks Reverb - URL Method.....	176
Indexing WebWorks Reverb - Client-Side Search for Source Documents, Baggage Files and External URLs.....	177
Using Tidy for Indexing HTML Pages.....	178
Assigning Relevance Weight to Your Source Documents Styles.....	179
Assigning Relevance Weight to Your HTML and PDF Baggage Files.....	180
Search Highlighting in Baggage Files - Client-Side Search.....	182
URL Toggle in WebWorks Reverb.....	183
End-user requirements in WebWorks Reverb.....	184
Wiki Markup.....	185
XML+XSL.....	186
Designing Input Format Standards.....	188
Designing Adobe FrameMaker Formats and Standards.....	189
Creating Standards to Support Single-Sourcing.....	190
Planning for Importing Elements Across Files.....	191
Paragraph Formats in FrameMaker.....	192
Character Formats in FrameMaker.....	195
Bulleted and Numbered Lists in FrameMaker.....	197
Image Formats and Considerations in FrameMaker.....	198
Table Formats in FrameMaker.....	200

Cross Reference Formats in FrameMaker.....	201
Markers in FrameMaker.....	202
Variables and Conditions in FrameMaker.....	205
Page Layouts in FrameMaker.....	206
Reference Pages, Table of Contents, and Indexes in FrameMaker.....	207
Automation and Scripting in FrameMaker.....	208
Designing Microsoft Word Templates and Standards.....	209
Word Standards to Support Single-Sourcing.....	210
Understanding the Microsoft Word Template File.....	211
Creating a Clean Base Template File.....	212
Paragraph Styles in Word.....	213
Character Styles in Word.....	215
Bulleted and Numbered Lists in Word.....	217
Bulleted Lists in Word.....	218
Numbered Lists in Word.....	219
Image Styles and Considerations in Word.....	220
Table Styles in Word.....	222
Field Codes.....	223
AutoText, AutoCorrect, and User-Defined Hotkeys.....	226
Toolbars and Menus in Word.....	227
Variables and Conditions in Word.....	228
Page Layouts and Sections in Word.....	229
Table of Contents and Index in Word.....	230
Automation with Macros in Word.....	231
Designing DITA Usage Standards.....	232
Using DITA Standards to Support Single-Sourcing.....	233
Mapping DITA Classes to ePublisher Styles.....	234
Defining Online Features with DITA.....	236
Customizing the DITA DTD.....	237
DITA Specialization.....	238
Designing, Deploying, and Managing Stationery.....	240
Checklist: Design, Deploy, and Manage Stationery.....	241
Prerequisite Tasks.....	242
Stationery Design Tasks.....	243
Optional Online Feature and Customization Tasks.....	244
Stationery Testing and Deployment Tasks.....	245

Understanding Stationery.....	246
Stationery Components.....	247
Understanding Stationery Synchronization.....	248
Designing Stationery.....	249
Creating a Stationery Design Project.....	250
Adding Output Formats to Your Stationery Design Project.....	251
Adding a Target to Your Stationery Design Project.....	252
Selecting an Active Target in Your Stationery Design Project.....	253
Updating a Project to Include All Styles.....	254
Understanding Style Designer.....	255
Modifying Output with CSS Properties and Attributes.....	256
Understanding the CSS Box Model.....	257
Inheriting Style Properties.....	258
Understanding Options in Style Designer.....	259
Organizing and Managing Styles.....	260
Previewing the Output from a Source File.....	261
Defining New Pages (Page Breaks).....	262
Defining TOCs and Mini-TOCs.....	263
Defining the Table of Contents Structure (Levels).....	264
Generating and Naming the Table of Contents File.....	266
Defining the Table of Contents from an Irregular Heading Hierarchy.....	267
Understanding the Table of Contents and Merge Settings.....	270
Defining the Icon for a Table of Contents Entry.....	271
Creating a Miniature Table of Contents.....	272
Modifying the Appearance of Mini-TOC Entries.....	273
Modifying the Appearance of Paragraphs.....	275
The Prototype Style for Paragraphs.....	276
Setting the Background Color of a Paragraph.....	277
Setting the Border Style and Color of a Paragraph.....	278
Setting the Font for a Paragraph.....	279
Setting the Width, Height, and Positioning of a Paragraph.....	280
Adjusting the Space Around a Paragraph.....	281
Setting the Text Color and Other Characteristics of a Paragraph.....	282
Modifying Paragraphs for Bidirectional Languages.....	283
Disabling Autonumbering in Output.....	284
Defining the Appearance of Notes, Tips, Cautions, and Warnings.....	285
Adjusting Paragraph Styles for Wiki Markup.....	286

Defining the Appearance of Bulleted Lists.....	287
Defining the Appearance of Numbered Lists.....	288
Defining the Appearance of Bulleted and Numbered Lists for Wiki Markup.....	289
Modifying the Appearance of Characters.....	290
The Prototype Style for Characters.....	291
Setting the Background Color of a Character.....	292
Setting the Border Style and Color of Characters.....	293
Setting the Font for a Character.....	294
Adjusting the Space Around Characters.....	295
Setting the Color and Other Characteristics of Characters.....	296
Modifying Characters for Bidirectional Languages.....	297
Defining the Appearance of Tables.....	298
The Prototype Style for Tables.....	299
Setting the Background Color or Image of a Table.....	300
Setting the Border Style and Color of a Table.....	301
Setting the Width and Height of a Table.....	302
Setting the Vertical and Horizontal Alignment within a Table.....	303
Adjusting the Space Within and Around a Table.....	304
Modifying Header, Footer, and Body Rows of a Table.....	305
Modifying Cells of a Table.....	306
Defining the Appearance of Images.....	307
Supported Image Formats.....	308
Image Quality and Processing.....	309
The Prototype Style for Images.....	310
Defining Graphic Styles.....	311
Setting the Border Style and Color of an Image.....	312
Modifying the Width, Height, and Positioning of an Image.....	313
Adjusting the Space Around Images.....	314
Using Thumbnails for Images.....	315
Setting the Maximum Width and Height for Images.....	316
Modifying Image Size by Scale.....	317
Modifying Image Resolution.....	318
Setting Color Bit Depth.....	319
Choosing an Image File Format and Quality Level.....	320
Creating Grayscale Images.....	321
Setting Transparency for .gif and .png Images.....	322
Defining the Appearance of Pages.....	323

The Prototype Style for Pages.....	324
Displaying Company Logo and Information on a Page.....	325
Modifying the Appearance of the Company Information.....	326
Setting the Background Color or Image of a Page.....	328
Setting the Border Style and Color of a Page.....	329
Adjusting the Space Around a Page.....	330
Using a Custom CSS to Modify the Appearance of Content.....	331
Modifying the Location and Separators of Breadcrumbs.....	332
Modifying the Appearance of Breadcrumbs.....	333
Choosing the Location of Navigation Browse Buttons.....	334
Modifying the Navigation Browse Buttons.....	335
Associating a Page with a Page Style.....	336
Defining the Appearance of Links.....	337
Saving a Snapshot (Backup Copy) of Your Project.....	338
Defining the Processing of Markers and Field Codes.....	339
Defining File Names.....	342
Specifying File Names for Pages Using Page Naming Patterns.....	343
Specifying File Names for Images Using Graphic Naming Patterns.....	345
Using Markers to Define File Names.....	347
Defining Context-Sensitive Help Links.....	348
Defining Filename Markers for Context-Sensitive Help Links.....	349
Defining TopicAlias Markers for Context-Sensitive Help Links.....	350
Defining Expand/Collapse Sections (Drop-Down Hotspots).....	351
Using Styles and Markers for Expand/Collapse Sections.....	352
Modifying Images for Expand/Collapse Sections.....	353
Defining Popup Windows.....	354
Using Marker Styles to Create Popup Windows.....	355
Using Paragraph Styles To Create Popup Windows.....	356
Assigning a Page Style to Popup Windows.....	357
Defining Related Topics.....	358
Using a Paragraph Style for Related Topics Lists.....	359
Defining See Also Links.....	360
Enabling See Also Functionality.....	361
Modifying the Appearance of the See Also Button.....	362
Define the Default Settings for Each Target.....	363
Defining the Default Index Settings.....	364
Defining the Default Processing of Variables.....	365

Defining the Default Processing of Conditions.....	366
Defining the Default Processing of Cross References.....	367
Defining Default PDF Generation Settings.....	368
Defining the Accessibility Report to Validate Content.....	369
Defining Other Reporting Options.....	371
Saving and Testing Stationery.....	372
Customizing Your Design.....	373
Understanding Customized Processing.....	374
Disabling External CSS Files.....	375
Mark of the Web.....	377
Customizing the Content Page Design.....	379
Customizing Output Format-Specific Features.....	380
Backing Up Your Stationery Design Project, Stationery, and Projects.....	381
Deploying Stationery.....	382
Managing and Updating Stationery.....	383
Customizing Stationery for Output Format-Specific Features.....	386
Customizations Specific to WebWorks Help.....	387
Renaming the Top-Level Entry File.....	388
Selecting a Theme.....	389
Customizing the Splash Page in WebWorks Help.....	390
Replacing the Splash Image.....	391
Removing the Splash Page.....	392
Customizing the Toolbar in WebWorks Help.....	393
Adding and Removing Toolbar Buttons in WebWorks Help.....	394
Replacing the Toolbar Buttons in WebWorks Help.....	395
Changing the Background Color of the Toolbar.....	396
Customizing the Navigation Pane in WebWorks Help.....	397
Setting the Initial Width of the WebWorks Help Navigation Pane.....	398
Controlling the Navigation Pane Hover Text Appearance.....	399
Changing the Font Color on the Navigation Pane Tabs in WebWorks Help.....	401
Using Custom Icons on the Contents Tab in WebWorks Help.....	402
Modifying the Appearance of the Search Message in WebWorks Help.....	403
Modifying the Search Ranking.....	405
Modifying the Search Highlighting.....	407
Synonyms.....	408
Minimum word length & common words.....	409

Using Context-Sensitive Help in WebWorks Help.....	410
Understanding Mapping Files in WebWorks Help.....	411
Opening Context-Sensitive Help in WebWorks Help using Standard URLs.....	412
Opening Context-Sensitive Help with the WebWorks Help API.....	413
Opening Context-Sensitive Help with the Javascript API.....	414
Customizations Specific to WebWorks Reverb.....	415
Changing the Appearance of WebWorks Reverb.....	416
Format Settings for WebWorks Reverb.....	418
WebWorks Reverb Format Settings.....	419
Google Format Settings.....	420
Social Format Settings.....	421
Selecting an Alternate Skin for WebWorks Reverb.....	422
Customizing the Top-Level Entry File.....	423
Specifying the Entry Page Name.....	423
Specifying the Entry Page Style.....	424
Customizing the Splash Page in WebWorks Reverb.....	425
Specifying the Splash Page Style.....	426
Replacing the Splash Image.....	427
Modifying the Splash Page.....	428
Removing the Splash Page.....	429
Using Context-Sensitive Help in WebWorks Reverb.....	430
Understanding Mapping Files in WebWorks Reverb.....	431
Opening Context-Sensitive Help in WebWorks Reverb using Standard URLs.....	432
URL Commands Support by WebWorks Reverb.....	433
Opening Context-Sensitive Help in WebWorks Reverb using Javascript.....	434
Opening Context-Sensitive Help in WebWorks Reverb using the WebWorks Help API.....	435
Configuring Client-Side Search for Reverb.....	436
Configuring Synonyms.....	437
Configuring Microsoft IIS Search for Reverb.....	438
Searching WebWorks Reverb - URL Method.....	439
Incorporating Google Analytics for Your Reverb Files.....	440
Steps to Create a Google Analytics Account.....	441
Steps to Obtain Your Google Analytics Account Identifier.....	442
Google Analytics behind a Firewall or Non-public Website.....	443
Configuring Commenting and End-User Feedback for Reverb.....	444
Steps to Create Your First Disqus Site.....	445
Customizations Specific to WebWorks Reverb 2.0.....	446

Changing the Appearance of WebWorks Reverb 2.0.....	447
Using SASS To Customize WebWorks Reverb 2.0.....	448
Understanding SASS Variables In WebWorks Reverb 2.0.....	450
Layout Colors In WebWorks Reverb 2.0 Skins.....	452
Format Settings for WebWorks Reverb 2.0.....	463
WebWorks Reverb 2.0 Format Settings.....	464
WebWorks Reverb 2.0 Toolbar Format Settings.....	465
WebWorks Reverb 2.0 Menu Format Settings.....	466
WebWorks Reverb 2.0 Page Format Settings.....	467
WebWorks Reverb 2.0 Footer Format Settings.....	468
Google Format Settings.....	469
Social Format Settings.....	470
Selecting an Alternate Skin for WebWorks Reverb 2.0.....	471
Customizing the Top-Level Entry File.....	472
Specifying the Entry Page Name.....	472
Specifying the Entry Page Style.....	473
Customizing the Splash Page in WebWorks Reverb 2.0.....	474
Specifying the Splash Page Style.....	475
Replacing the Splash Image.....	476
Modifying the Splash Page.....	477
Removing the Splash Page.....	478
Using Context-Sensitive Help in WebWorks Reverb 2.0.....	479
Understanding Mapping Files in WebWorks Reverb 2.0.....	480
Opening Context-Sensitive Help in WebWorks Reverb 2.0 using Standard URLs.....	481
URL Commands Support by WebWorks Reverb 2.0.....	482
Opening Context-Sensitive Help in WebWorks Reverb 2.0 using JavaScript.....	483
Opening Context-Sensitive Help in WebWorks Reverb 2.0 using the WebWorks Help API.....	484
Configuring Client-Side Search for Reverb 2.0.....	485
Configuring Synonyms.....	486
Searching WebWorks Reverb 2.0 - URL Method.....	487
Incorporating Google Analytics for Your Reverb 2.0 Files.....	488
Configuring Commenting and End-User Feedback for Reverb 2.0.....	489
Steps to Create Your First Disqus Site.....	490
Customizations Specific to Microsoft HTML Help.....	491
Adjusting the HTML Help Viewer Window Size and Toolbar Buttons.....	492
Creating an Additional HTML Help Window Definition.....	493
Using Context-Sensitive Help in HTML Help.....	494

Understanding Mapping Files in HTML Help.....	495
Setting Mapping File Options for HTML Help.....	497
Testing Context-Sensitive HTML Help.....	498
Defining What's This (Field-Level) Help in HTML Help.....	499
Customizations Specific to Simple HTML.....	500
Removing the Left and Right Margins from Simple HTML Pages.....	501
Customizations Specific to Dynamic HTML.....	502
Using SASS to change the Appearance of Dynamic HTML.....	503
Modifying the Appearance of the Table of Contents in Dynamic HTML.....	504
Modifying the Appearance of the Index in Dynamic HTML.....	506
Other Changes to Text in the TOC and Index in Dynamic HTML.....	508
Customizations Specific to Oracle Help and Sun JavaHelp.....	509
Defining the Navigation Pane in Oracle Help.....	510
Using Custom Windows in Oracle Help.....	511
Defining the Navigation Pane in Sun JavaHelp.....	512
Using Context-Sensitive Help in Oracle Help and Sun JavaHelp.....	513
Understanding Mapping Files in Oracle Help and Sun JavaHelp.....	514
Testing Context-Sensitive Oracle Help and Sun JavaHelp.....	515
Customizations Specific to Eclipse Help.....	516
Using Markers to Specify Context Plug-ins in Eclipse Help.....	517
Using Markers to Specify Topic Descriptions for Context-Sensitive Help Topics in Eclipse Help.....	518
Customizations Specific to Wiki Markup.....	519
Defining Wiki Categories or Labels.....	520
Defining Table File Naming Patterns for Wiki - MoinMoin Output.....	521
Understanding Default File Naming for Tables in Wiki - MoinMoin Output.....	522
Specifying Table File Naming Patterns for Wiki - MoinMoin Output.....	523
Designing Wiki Output.....	525
Why Use a Wiki to Deliver Content?.....	526
Checklist: Preparing to Deliver Content on a Wiki.....	528
Identifying Wiki Content Users.....	529
Identifying Your Wiki Server and Wiki Output Format.....	530
Defining the Wiki Content Maintenance Process.....	531
Reviewing and Managing Changes to Wiki Content.....	532
Incorporating Changes Back Into Source Documents.....	533
Encouraging Content Contributors to Contribute.....	534

Defining Wiki Navigation.....	535
Understanding Wiki Search Capabilities.....	536
Creating Landing Pages.....	537
Defining and Configuring Wiki Permissions.....	538
Verifying Your Wiki Server Configuration.....	539
Generating and Deploying Wiki Content.....	540
Wiki Deployment - Start to Finish.....	542
Wiki WYSIWYG Support.....	544
Integrating ePublisher Online Help with the WebWorks Documentation Wiki.....	545
Designing PDF Output.....	548
Why use PDF - XSL-FO output format?.....	549
PDF XSL-FO Font Inclusions.....	550
Custom Header and Footers.....	552
Designing ePUB Output.....	554
ePUB Platforms.....	555
ePUB Considerations.....	556
Meta Data.....	557
Book Title and ID.....	558
Long Content.....	559
Page Styles.....	560
Tables.....	561
Cover.....	562
Syncing with Apple iPad.....	563
Producing Output Based on Stationery.....	565
Checklist: Producing Output Based on Stationery.....	566
Understanding Projects and the Project Folder Structure.....	567
Understanding Projects.....	568
Understanding the Project Folder Structure.....	569
Understanding Source Documents.....	571
Understanding Baggage Files.....	572
Understanding Targets.....	573
Understanding Stationery.....	574
Creating Projects Based on Stationery.....	575
Working with Source Documents.....	576
Adding Source Documents to Projects.....	577

Opening Source Documents from Document Manager.....	579
Scanning Source Documents.....	580
Understanding Scanning and Scanning Options.....	581
Setting Scanning Options.....	582
Scanning Selected Documents.....	583
Scanning All Documents.....	584
Relinking Source Documents.....	585
Removing Source Documents from Projects.....	586
Understanding Source Documents Groups.....	587
Organizing Source Documents Using Groups.....	588
Creating Top-Level Groups.....	589
Creating Subgroups.....	590
Renaming Groups.....	591
Rearranging Source Documents in Groups.....	592
Removing Groups.....	593
Working with Targets.....	594
Specifying Active Targets.....	595
Adding Targets to Projects Based on Stationery.....	596
Renaming Targets.....	597
Deleting Targets.....	598
Working with Projects.....	599
Saving Projects.....	600
Opening Existing Projects.....	601
Closing Projects.....	602
Synchronizing Projects with Stationery.....	603
Understanding Manifest Files.....	604
Understanding Stationery Files.....	605
When to Synchronize.....	606
Automatically Synchronizing ePublisher Express Projects with Stationery.....	607
Manually Synchronizing ePublisher Express Projects with Stationery.....	608
Deleting Projects.....	609
Generating and Regenerating Output.....	610
Understanding Output Generation and Regeneration.....	611
Generating Output.....	612
Regenerating Output.....	613
Generating Output from FrameMaker or Microsoft Word.....	614
Modifying Help System Title Bars.....	615

Customizing or Removing Splash Page Images in WebWorks Help.....	616
Customizing Splash Page Images in WebWorks Help.....	617
Removing Splash Page Images in WebWorks Help.....	618
Viewing Output.....	619
Viewing Output by Automatically Opening Generated Output.....	620
Viewing Output in Output Explorer.....	621
Viewing Output in the Output Folder.....	624
Changing the Location of the Output Folder.....	625
Working with Output Log Files.....	626
Validating Output Using Reports.....	627
Understanding Accessibility Reports.....	628
Understanding Baggage Files Reports.....	629
Understanding Conditions Reports.....	630
Understanding Filenames Reports.....	631
Understanding Links Reports.....	632
Understanding Styles Reports.....	633
Understanding Topics Reports.....	634
Understanding Images Reports.....	635
Understanding Printable Reports.....	636
Configuring Reports.....	637
Generating Reports.....	639
Understanding Report Messages.....	640
Accessibility Report Messages.....	641
Baggage Files Report Messages.....	642
Filename Report Messages.....	643
Links Report Messages.....	644
Topics Report Messages.....	645
Images Report Messages.....	646
Merging Help Systems (Multivolume Help).....	647
Deploying Output.....	650
Understanding Output Deployment.....	651
Creating Output Destinations.....	652
Specifying Output Destinations for Targets.....	654
Deploying Output to Output Destinations.....	655
Customizing Target Settings.....	656
Specifying Accessibility Settings.....	658
Specifying Baggage Files Settings.....	659

Baggage files info list.....	660
Copy baggage file dependents.....	662
Create standalone group.....	663
Index baggage files.....	664
Index external links.....	665
Standalone group name.....	666
Specifying Company Information.....	667
Specifying File Processing Behavior for Front Matter, Index, and Table of Contents Files.....	668
Specifying Page Breaks Settings.....	669
Specifying Page, Image, and Table File Naming Patterns.....	670
Specifying Index Settings.....	671
Specifying How Links to Files or External URLs Display in Browser Windows.....	672
Specifying Character Encoding for Targets.....	673
Specifying the Language Used by Targets.....	674
Specifying PDF Generation Settings.....	675
Specifying Table of Contents Settings.....	676
Specifying Report Settings.....	678
Specifying Output Format-Specific Settings.....	679
Customizing Variable Settings in Projects.....	680
Customizing Condition Settings in Projects.....	681
Customizing Cross-Reference Settings in Projects.....	682
Modifying Cross-Reference Formats in Projects.....	683
Adding Cross-Reference Formats to Projects.....	684
Deleting Cross-Reference Formats from Projects.....	685
Customizing File Mappings.....	686
Understanding File Mappings.....	687
Modifying File Mappings.....	688
Creating New File Mappings.....	689
Deleting File Mappings.....	690
Scheduling and Integrating Processes with AutoMap.....	692
How ePublisher Supports Automation.....	693
What Is ePublisher AutoMap?.....	694
Benefits of Using ePublisher AutoMap.....	695
Version Control System (VCS) Integration.....	696
Content Management System (CMS) Integration.....	697
Preparing Projects, Stationery, and Source Files.....	698

Starting ePublisher AutoMap.....	699
Setting ePublisher AutoMap Preferences.....	700
Specifying the Job, Staging, and User Formats Folder Locations.....	701
Job Folder.....	702
Staging Folder.....	703
User Formats Folder.....	704
Automatic Scanning for Conditions and Variables.....	705
Keeping or Deleting Temporary Files.....	706
Defining File Mappings.....	707
Defining Output Destinations.....	708
Defining Email Notifications.....	709
Selecting Console Language (English, German, French, and Japanese).....	710
Working with Jobs.....	711
Creating a Project-Based Job.....	712
Creating a Stationery-Based Job.....	713
Duplicating an Existing Job.....	715
Editing an Existing Job.....	716
Scheduling Jobs with Windows Scheduler.....	717
Deleting an Existing Schedule for a Job.....	718
Running an Existing Job.....	719
Viewing a Job Log File.....	720
Canceling a Job.....	721
Deleting an Existing Job.....	722
Using Scripts for Additional Custom Processing.....	723
Writing Scripts.....	724
Working Folder.....	725
Opening and Using the Script Editor.....	726
Scripting Variables.....	727
Scripting Examples.....	728
Show Time and Date Example.....	729
Using Scripting Variables Example.....	730
CVS Version Control Checkout Example.....	732
Using the Command-Line Interface.....	734
Running ePublisher AutoMap from the Command Line.....	735
CLI Syntax and Reference.....	736
CLI Examples.....	738
Running a Project and Updating the Express project file.....	739

Running a Project and Generating Only One Target.....	740
Running a Project from Scratch and Deploying to a Clean Location.....	741
Running a Project and Deploying to an Alternate Location.....	742
Running a Job Without Sending Notification When Done.....	743
Running a Job and Deploying to a Clean Location.....	744
Running a Job Without Deploying the Content.....	745
Understanding How ePubublisher Works.....	747
Understanding Terminology.....	748
Understanding the Processing Workflow.....	749
Understanding the Transformation Process.....	750
Adapters Transform Source Documents to WIF.....	751
Understanding WebWorks Intermediate Format (WIF).....	752
Processing Files by Type.....	753
Identifying Files to Process.....	754
TOC Processing Example.....	755
Understanding Stationery, Projects, and Overrides.....	756
File Locations.....	757
File Processing.....	758
ePublisher File Types.....	759
Format Trait Info (*.fti) Files.....	760
format.wwfmt Files.....	761
files.info Files.....	762
Stationery Design Project .wep File.....	763
Project .wrp File.....	764
Stationery .wxsp File.....	765
XSL Match Templates.....	766
Root Match Templates.....	767
Root Match Templates in ePubublisher.....	768
Extension Objects.....	769
Output and Console Customizations.....	770
Project Format and Target Overrides.....	771
Creating Format Overrides.....	772
Creating Target Overrides.....	774
Managing Format/Target Overrides.....	776
Customizing Page Templates.....	781
Namespace and Attributes.....	782

Using ePubublisher Style Variables in Page Templates.....	785
Features - From Input to Output.....	787
Popups.....	788
Using Paragraph Styles.....	789
Input Instructions.....	789
ePublisher Instructions.....	790
Using Markers.....	791
Input Instructions.....	791
ePublisher Instructions.....	792
Available Outputs.....	793
Related Topic.....	794
Using Paragraph Styles.....	795
Input Instructions.....	796
ePublisher Instructions.....	797
Available Outputs.....	798
See Also Links.....	799
Using Markers.....	800
Input Instructions.....	800
ePublisher Instructions.....	801
Available Outputs.....	802
Context Sensitive Help.....	803
Using Markers.....	804
Input Instructions.....	804
ePublisher Instructions.....	805
Available Outputs.....	806
Multi-volume Help.....	807
Using Groups.....	808
ePublisher Instructions.....	809
Available Outputs.....	810
Mini TOC.....	811
Using Paragraph Styles.....	812
Input Instructions.....	812
ePublisher Instructions.....	813
Available Outputs.....	814
Custom TOC icon.....	815
Using Makers.....	816

Input Instructions.....	816
ePublisher Instructions.....	817
Available Outputs.....	818
Wiki Categories and Labels.....	819
Using Markers.....	820
Input Instructions.....	820
Available Outputs.....	821
Page Styles.....	822
Using Markers.....	823
Input instructions.....	823
ePublisher instructions.....	824
Available Outputs.....	825

Copyright © 2001-2018 Quadralay® Corporation. All rights reserved.

Quadralay and WebWorks are registered trademarks of Quadralay Corporation. FrameMaker is a registered trademark of Adobe Systems, Inc. Microsoft and Windows are registered trademarks of Microsoft Corporation. Oracle is a registered trademark of Oracle Corporation. Sun, Java and JavaHelp are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. . Doc-To-Help is a registered trademark of ComponentOne LLC. All other trademarks used herein are the properties of their respective owners and are used for identification purposes only.

Portions of this software are copyrighted by others, as follows:

.NET Framework Checker NSIS plugin

All NSIS source code, plug-ins, documentation, examples, header files and graphics, with the exception of the compression modules and where otherwise noted, are licensed under the zlib/libpng license.

Apache

Copyright © 1999-2006 The Apache Software Foundation

DITA-OT

The DITA Open Toolkit is licensed for use, at the user's election, under the Common Public License v1.0 or the Apache Software Foundation License v2.0. If, at the time of use, the Project Management Committee has designated another version of these licenses or another license as being applicable to the DITA Open Toolkit, user may select to have its subsequent use of the DITA Open Toolkit governed by such other designated license.

Copyright © 1999-2006 The Apache Software Foundation

Flvplayer

The flvplayer is licensed for use under the Mozilla Public License (MPL) version 1.1.

Ghostscript API wrapper

Copyright © 2004, Pelagon Limited. All rights reserved.

HTML Tidy

Copyright (c) 1998-2016 World Wide Web Consortium (Massachusetts Institute of Technology, European Research Consortium for Informatics and Mathematics, Keio University). All Rights Reserved.

Additional contributions (c) 2001-2016 University of Toronto, Terry Teague, @geoffmcl, HTACG, and others.

IKVM

Copyright © 2002-2011 Jeroen Frijters All rights reserved.

ImageMagick

Copyright © 2002 ImageMagick Studio, a non-profit organization dedicated to making software imaging solutions freely available.

International Components for Unicode (ICU)

Copyright © 1995-2003 International Business Machines Corporation and others. All rights reserved.

iText

Copyright © 1999-2005 by Bruno Lowagie and Paulo Soares. All rights reserved.

jQuery

Copyright © jQuery Foundation.

Java and JavaHelp

Copyright © 2003 Sun Microsystems, Inc. All rights reserved. Use is subject to license terms. Sun, Sun Microsystems, the Sun Logo, Solaris, Java, the Java Coffee Cup Logo, JDK, Java Foundation Classes (J.F.C.), Java Plug-in and JavaHelp are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

JGL - The Generic Collection Library for Java™

Copyright © 1997 ObjectSpace, Inc. All Rights Reserved.

Microsoft DotNetZip

Copyright © 2006, 2007 Microsoft Corporation, One Microsoft Way, Redmond, Washington 98052-6399 U.S.A. All rights reserved.

Microsoft Help Workshop

Copyright © 2004 Microsoft Corporation, One Microsoft Way, Redmond, Washington 98052-6399 U.S.A. All rights reserved.

Mistune

Copyright (c) 2014 - 2015, Hsiaoming Yang. All rights reserved.

NetAdvantage

Copyright ©1992-2005 Infragistics, Inc., Windsor Corporate Park, 50 Millstone Road, Building 200 - Suite 150, East Windsor, NJ 08520. All rights reserved.

Newtonsoft.Json

Copyright (c) 2007 James Newton-King.

NSIS (Nullsoft Scriptable Install System)

Copyright (C) 1999-2017 Contributors.

All NSIS source code, plug-ins, documentation, examples, header files and graphics, with the exception of the compression modules and where otherwise noted, are licensed under the zlib/libpng license.

nsisDDE plugin

Copyright (c) 2005-2010 Olivier Marcoux.

Oracle Help

Copyright © 2002 Oracle Corporation. All Rights Reserved.

Python

The Python programming language by Guido van Rossum, PythonLabs, and many contributors -- Python Software Foundation License version 3

The Python Software Foundation License version 3 is an OSI Approved Open Source license. It consists of a stack of licenses that also include other licenses that apply to older parts of the Python code base. All of these are included in the OSI Approved license: PSF License, BeOpen Python License, CNRI Python License, and CWI Python License. The intellectual property rights for Python are managed by the Python Software Foundation.

Saxon

Copyright © 1999-2003 Intalio, Inc. All Rights Reserved.

Various .Net Utilities

Portions copyright © 2002-2004 The Genghis Group (<http://www.genghisgroup.com/>)

Xerces

Copyright © 1999-2004 The Apache Software Foundation. All rights reserved.

WebWorks ePublisher Documentation

WebWorks ePublisher Documentation provides many resources online and in the product to help you use the ePublisher system to deliver integrated, single-sourced document sets.

[Dockable Help](#)

Provides context-sensitive information and step-by-step guidance for common tasks, as well as descriptions of each field on each window. Dockable Help is the ePublisher documentation set in Reverb 2.0 format and provides a comprehensive, integrated deliverable that includes all the information from WebWorks resources. Dockable Help displays when ePublisher is opened, or you can click **Documentation** on the **Help** tab in the product interface.

Contacting Quadralay

Please contact us with your questions and comments. We look forward to hearing from you. If you need assistance with an issue, please contact Technical Support. The Support Web site allows you to review the support policy and create a case to track an issue.

Telephone: 877.893.2967 (only in the United States and Canada)
512.719.3399

Email: info@webworks.com

Support www.webworks.com/Support/

Web Site: www.webworks.com

Conventions

WebWorks ePublisher documentation uses consistent conventions to help you identify items. The following table summarizes these conventions.

Convention	Use
Bold	<ul style="list-style-type: none">Window and menu itemsTechnical terms, when introduced
<i>Italics</i>	<ul style="list-style-type: none">Book titlesVariable namesEmphasized words
Fixed Font	<ul style="list-style-type: none">File and folder namesCommands and code examplesText you must typeText (output) displayed in the command-line interface
Blue	<ul style="list-style-type: none">Links
>	<ul style="list-style-type: none">Submenu selections, such as Target > Target Settings
Brackets, such as [value]	<ul style="list-style-type: none">Optional parameters of a command
Braces, such as {value}	<ul style="list-style-type: none">Required parameters of a command
Logical OR, such as value1 value2	<ul style="list-style-type: none">Exclusive parameters. Choose one parameter.

1

Introduction to the WebWorks ePublisher Platform

Content development, publication, and maintenance are complex, time-consuming processes. Each day, companies spend numerous hours writing, formatting, and publishing information needed by internal and external users. At the same time, these users search among vast amounts of content to find the information they want and need. Companies need to streamline content production processes while delivering the content to users when, where, and how they need it.

Adding to this complexity, teams across the company use multiple content authoring tools, such as Adobe FrameMaker, Microsoft Word, and XML editors, to create the content. These teams must be able to use the authoring tools that best meet their needs. However, mastering these tools for content development is only half the battle. Content developers also need tools to publish content consistently in multiple formats, such as print, HTML, and PDF. This requirement is difficult to meet and often leads to increased production costs or an inconsistent corporate image. In addition, corporate branding standards change over time, and implementing these changes across all content adds to production and maintenance costs.

With all these variations in content creation, publication, and maintenance, delivering the right information to the right people in the right format and at the right time is an endless and costly struggle that consumes enormous time and resources across organizations.

What Is ePublisher?

The WebWorks ePublisher Platform (ePublisher) is a powerful, comprehensive solution that delivers cost-effective processes for efficiently publishing and maintaining online and print information. ePublisher gives you the flexibility to deliver content from multiple types of source documents, such as Adobe FrameMaker, Microsoft Word, and DITA, in virtually any output format you need without incurring training or software deployment expenses. The open, standards-based architecture provides a powerful engine that does not lock your content in a proprietary format that can become outdated as tools and standards change.

With the robust combination of input and output formats supported by ePublisher, you can develop the content using your preferred content authoring tools, and then produce and maintain all your deliverables within a single publishing environment. You can implement a consistent look and feel across all deliverables and quickly modify and deploy that branding if and when needed. ePublisher integrates seamlessly with your content management or version control systems, so you can automatically generate and deploy the deliverables you need and reduce the time demands on your teams.

Workflow

WebWorks ePublisher Platform components provide a workflow that ensures you can deliver your content, your way, every time. A successful online content delivery workflow includes the following items:

- Creation and automation of consistent, reusable online content designs
- Packaging of online content designs for seamless, consistent reuse
- Application of online content designs to new and existing projects to ensure consistent content delivery and deployment

ePublisher supports this workflow by allowing ePublisher users to perform the following tasks:

- Stationery designers use ePublisher Designer to create and manage online content designs, then package the designs into Stationery for writers to use
- Writers use ePublisher Express and Stationery to create and deploy consistent online content
- ePublisher AutoMap can be configured to automatically generate and deploy online content

For more information about the WebWorks ePublisher Platform components that support this workflow, see “WebWorks ePublisher Platform Components”. For more information about the ePublisher workflow, see “Understanding the ePublisher Workflow”.

WebWorks ePublisher Platform Components

With ePublisher, you can write your content in your preferred authoring tools, then use ePublisher components to design and deliver your content. The ePublisher components allow you to design all the content output formats you need, and then automate the publication process and integrate it with your company-wide processes, such as product builds and Web site updates.

ePublisher includes the following components:

ePublisher Designer

The design tool for creating and designing Stationery. **Stationery** defines the appearance and functionality of all the output formats you need. ePublisher provides several default formats that you can use as a basis for your Stationery, and then you can customize that standard and save it as your Stationery for your deliverables produced using the other ePublisher components.

ePublisher Express

The on-demand publishing tool that transforms your content based on your Stationery and converts your source documents into the desired output formats. This component is installed on the desktop and integrates with your existing authoring tools to support the features you require, such as related topics and expand/collapse sections within your deliverable. With this component, you can quickly prepare your source documents and generate your final deliverables.

ePublisher AutoMap

The automation tool that enables you to automate the content conversion process, batch processing, and integration with content management or version control systems. This component lets you schedule conversion projects to occur at times when you are not using your computer. For example, you can schedule the conversion to occur overnight. Then, when you arrive the next morning, your transformed content is ready for you. You can also automatically generate and deploy deliverables to meet your specific needs, such as updating Web site content based on updated source documents.

Supported Input Formats

ePublisher provides a single-sourcing environment that works with the following input formats.

- Adobe FrameMaker
- Microsoft Word
- DITA XML Files
- WebWorks Markdown

Supported Output Formats

ePublisher lets you define your **output formats**, such as XHTML and WebWorks Help, so content developers can focus on developing quality content without worrying about tedious conversion requirements for various deliverables. You can manage your content as you want and produce deliverables in the following formats:

- WebWorks Reverb 2.0
- WebWorks Reverb
- WebWorks Help 5.0
- Dynamic HTML
- Microsoft HTML Help
- Eclipse Help
- Sun JavaHelp 2.0
- Oracle Help
- PDF
- PDF - XSL-FO
- eBook - ePUB 2.0
- XML+XSL

How ePublisher Helps You

With ePublisher and its agile enterprise publishing capabilities, you have unparalleled design flexibility with the ability to deliver your information, regardless of input format, in multiple output formats. This solution enables both large and small organizations to implement the publishing environment that works best for them.

Streamline and Automate the Content Publishing Process

In a traditional content authoring environment, a content author produces content designed for a single output format. This environment typically has the following limitations:

- Content is often duplicated across multiple content-producing teams.
- Content is not maintained consistently across the multiple teams.
- Production is expensive with multiple tools and technologies.

Using ePublisher, you can quickly publish content from your source documents. You can develop the content using your preferred content authoring environment, such as Microsoft Word, Adobe FrameMaker, or DITA. You do not have to spend time and resources learning how to format content for each and every output format in which your content will be delivered. You can focus on the content, and then use your Stationery to quickly and easily deliver information in the multiple formats that meet the requirements for your organization.

ePublisher reduces wasted time and expenses that occur when multiple groups in your organization unknowingly produce the same information at the same time. This publishing environment allows your organization to produce content one time. This content can then be shared in varying input formats across your organization.

Produce High Quality Deliverables with Fewer Individual Dependencies

Instead of requiring team members to understand the entire process to produce content in multiple formats, team members can focus on their areas of expertise. Content developers can create informative content to add value to the products and services they document. You can also expand the skills of individual team members into important new areas, which strengthens your team as a whole.

ePublisher lets you concentrate on producing high-quality content within the authoring environment that works for you. You spend much less time on designing, implementing, and delivering multiple output formats. With ePublisher, you quickly generate complete, ready-to-deploy publications.

ePublisher also allows you to preview, proof, and review your content before you publish and deploy it. The comprehensive reports and on-demand reporting help you identify and correct any issues that may effect your online content, such as invalid styles, missing links, and compliance with Web accessibility standards. In this way, ePublisher ensures that the content you produce is of the highest quality and consistency.

Reduce Support Costs and Increase Customer Satisfaction

When you consider the many steps content goes through to get from the content developer to its final destination, publishing content can be a tedious, costly process. When an organization does not commit its attention and resources to product information, the negative results impact many aspects of the business.

Customers want to find their solution by reading as little as possible. Consistent content helps customers skim the content and find the information they need. The time content developers spend formatting content for various output formats reduces the time they have to review and improve the content. Customers can become frustrated when they spend time sorting through inconsistent and potentially inaccurate or incomplete information.

Frustrated customers quickly give up and contact customer support, often with a negative impression of the company. Increased customer support calls, especially for basic concepts and product usage, can waste valuable company resources.

ePublisher provides a workflow designed to make the publishing process as non-intrusive as possible. By allowing you to choose your preferred authoring environment and having role-focused software components, training time and production costs are reduced. The published content is consistent and delivers more value to your customers.

Quickly Update and Deliver Content More Often

If your style or online content requirements change in your organization, you need to make changes throughout your content to implement these new requirements. Extended production times increase the difficulty and complexity of these changes. ePublisher streamlines the production process to enable you to quickly implement and deploy updated content. With these streamlined processes in place, you can deliver updated content more often.

ePublisher allows you to define and deploy centralized Stationery that all projects use. When corporate standards change, such as logos and branding, you can quickly update the Stationery to incorporate the new standards. Then, content developers can import the updated Stationery into their projects and publish their updated deliverables using the new standards without changing or redesigning their source documents.

Reduce Content Management Life Cycle Costs

Due to the limitations of the traditional content model, many organizations want to move to a single-sourcing environment. **Single sourcing** allows the same content to be used multiple times and delivered in different formats. Organizations use single sourcing to eliminate duplicate content, reduce content translation and maintenance costs, improve content consistency, and minimize errors. Single sourcing also allows organizations to produce information in various formats using the same source.

Many single-sourcing solutions require all content authors to use the same authoring tool. ePublisher allows you to develop the content using your preferred content authoring environments, such as Microsoft Word, Adobe FrameMaker, or DITA. Each department can standardize on the authoring tool that is right for them, and ePublisher ties all the input formats together with a single, unified, reliable publishing process. ePublisher allows you to create integrated deliverables with source documents from multiple authoring tools.

With ePublisher, you can use your existing authoring tools and content management systems to meet organization-wide publishing needs without incurring training or software deployment expenses. The open architecture, based on industry-standard XSL, provides a flexible solution that you can customize to meet your needs without locking you into a proprietary format that could result in expensive future migration costs.

How Organizations Use ePublisher

Companies use ePublisher to meet many of their content development, delivery, and maintenance needs:

- Update Web sites automatically with thousands of HTML pages every day
- Merge content across functional boundaries and deliver consistent content on corporate intranets and extranets
- Deliver integrated, context-sensitive help systems with products
- Single-source and deliver content in online and print formats
- Deploy content for multiple platforms and devices

The following sections highlight several ways you can use ePublisher to deliver consistent, comprehensive information.

Automatically Update Content on Web Sites and Wikis

Corporate Web sites have evolved into far more than just flashy advertising with contact information for your business. In addition to attention grabbing marketing about products and features, many company Web sites feature tutorials, product demos, specific product requirements and details, and Web 2.0 resources such as Wikis and community forums where customers can share information.

With ePublisher, you can consistently update the content on your Web site to maintain the latest information and make sure it is available to your customers. You can schedule and automate content processing and deployment to deliver up to date information each and every day.

ePublisher also allows you to easily maintain corporate intranets and publish source documents from multiple organizations across your company. You can define a standard Stationery and templates for teams to use. You can then define an ePublisher job and schedule it to search a drop-box folder on a regular basis and publish the content from the source documents in that folder using your standard Stationery. This scenario ensures your team members have the latest information they need and reduces the expenses associated with publishing and maintaining this content on your intranet.

Deliver Full-Featured, Context-Sensitive Help Systems

Products need to provide comprehensive help systems that meet the needs of many potential audiences. Content design and delivery must ensure that users get the information they need when, where, and how they need it. Some products need to deliver different content to different audiences. Other products are sold by multiple companies and require distinct product branding.

ePublisher provides comprehensive support for many advanced features used in online content design and delivery, including the following elements:

- Customizable browse navigation and breadcrumbs
- Customizable table of contents and mini-TOCs
- Pop-ups and expandable/collapsible text sections
- Related topics
- Images, image maps, and multiple forms of multimedia
- Context-sensitive help topics
- Merged help systems (multi-volume help)
- Variables and conditions
- Accessibility features, such as alternate text and long descriptions
- Field-level help

Produce Single-Sourced Print and Online Optimized Content

Customers have different needs and expectations for product content. In many cases, producing information in multiple formats for users involves extensive conversion and customization work to develop and deliver the various formats. Content authors must shift their attention to manipulating and converting the content into the many different user formats, often for both print and online, instead of focusing their time and efforts on developing quality information for users.

With ePublisher, you can quickly and efficiently produce consistent, effective print and online content in multiple formats. ePublisher provides XML/XSL processing and intelligent caching to process your source documents faster than ever before.

ePublisher produces the formatting code for you, whether it is HTML, XML, RTF, Wiki Markup, or a completely custom format. You do not need to know how to tag files for various output formats. With ePublisher, content developers can produce a printable PDF manual and a comprehensive online help deliverable immediately after finishing their content using the Stationery defined separately from their content.

Features Available in Each Output Format

ePublisher supports many output formats and you can implement many powerful features in your online content. Some features are available only in certain output formats. The following table summarizes which features are available in each output format.

Feature	Dyna- HTML	XML and XSL	Web Help	Web Rever-	Web HTML Help	Micro- Help	Soft- Help	Sun Java	Oracle Help	PDF	PDF - XSL- FO	eBook - ePUB 2.0	Wiki - Atlas	Wiki - Confluence	Wiki - MediaWiki	Wiki - Moin
Abbreviation alternate text	+		+	+	+				+		+					
Acronym alternate text	+		+	+	+				+		+					
Bidirectional language support	+	+	+	+	+	+					+					
Bullet customization	+		+	+	+						+					
Categories													+		+	+
Citation alternate text	+		+	+	+				+		+					
Context-sensitive help using topic aliases (topic IDs)			+	+	+	+	+	+	+							
CSS customization	+		+	+	+	+			+				+		+	
Expand/Collapse sections	+	+	+	+	+	+										
Favorites			+		+	+	+	+	+							
File name definition and control	+	+	+	+	+	+	+	+	+		+	+	+	+	+	+
Help window customization				+	+				+							
Image alternate text	+	+	+	+	+	+	+	+	+			+	+	+	+	+
Image long description	+	+	+	+	+	+	+	+	+			+	+	+	+	+
Image long description by reference	+		+	+	+	+	+	+	+			+	+	+	+	+
Image map alternate text	+	+	+	+	+	+	+	+	+			+				
Image scaling by image (GraphicScale)	+		+	+	+	+	+	+	+		+	+				
Image style by image (GraphicStyle)	+		+	+	+	+	+	+	+		+	+				
Index	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Inline comments													+			+
Meta tag keywords	+		+	+		+	+	+	+							
Mini (partial) table of contents	+		+	+	+	+	+	+	+		+		+	+	+	+
Multimedia links	+		+	+	+	+					+					

Feature	Dyna- mic HTML	XML and XSL	Web Help	Web Rever- se	Micro- HTML Help	Soft- scape Help	Sun Java Help	Oracle Help	PDF	PDF - XSL- FO	eBook - ePUB 2.0	Wiki - Atlas Confluence	Wiki - Media Wiki	Wiki - Win Moin
Page style customization	+	+	+	+	+	+	+	+		+	+	+	+	+
Popup windows			+		+		+	+						
Related topics	+		+	+	+	+	+	+				+	+	+
Search			+	+	+	+	+	+				+	+	+
See also links			+		+									
Splash page customization			+	+										
Table alternate text (table summary)	+		+	+	+			+						
Table of contents	+		+	+	+	+	+	+	+	+		+	+	+
Table of contents icon customization per topic			+		+		+	+						
Toolbar customization			+	+	+									
What's This help					+									

2

Planning and Installing ePublisher

This section helps you plan your ePublisher installation and install ePublisher components. This section provides information about ePublisher components and supported configurations and ePublisher requirements. This section also explains how to download and install ePublisher components, use your **contract identifier** (Contract ID), work with license keys, upgrade ePublisher, and troubleshoot installation and licensing issues.

Licensing Considerations

Before you can generate output using ePublisher, you must have a valid Contract ID. ePublisher uses your Contract ID to automatically handle the licensing of all ePublisher components and features. Your Contract ID is valid for your ePublisher use. Your contract ID may also be valid for other users, as long as the other users were included in the contract associated with the contract ID at the time ePublisher was purchased or the other users have been added to the same contract.

Currently ePublisher is licensed based on component and input format. ePublisher components include ePublisher Express, ePublisher Designer, and ePublisher AutoMap. ePublisher input formats include Adobe FrameMaker, Microsoft Word, and DITA-XML. Based on the input format of the files you use to author content, you may have access to one or more input formats. For more information about each ePublisher component, see “WebWorks ePublisher Platform Components”. For more information about Contract IDs, see “Working with Contract IDs”.

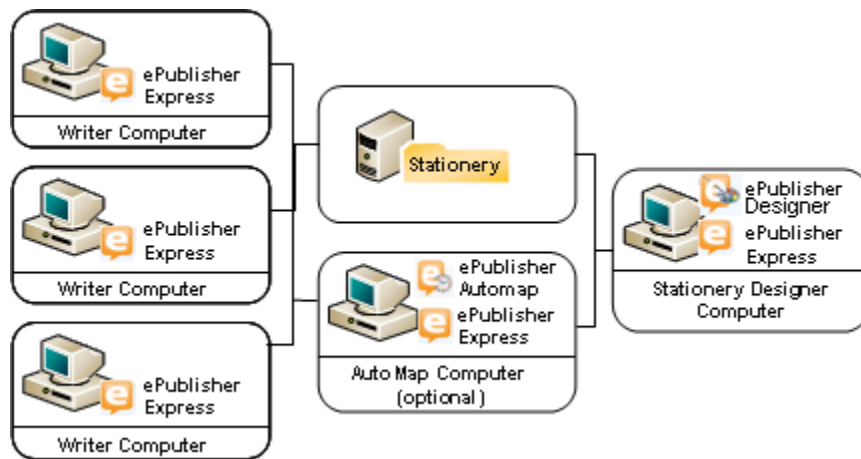
Components and Supported Configurations

Stationery designers must install ePublisher Designer and ePublisher Express on their computers. Stationery designers use ePublisher Designer to design Stationery and ePublisher Express to test Stationery.

Writers install ePublisher Express on their computers. Writers use ePublisher Express to generate output using Stationery created by a Stationery designer.

If you want to use AutoMap to automate output generation and integrate your output generation with content management or version control systems, install ePublisher AutoMap and ePublisher Express on the computer where you want to use ePublisher AutoMap. ePublisher AutoMap requires ePublisher Express. You can install ePublisher AutoMap on its own separate computer, or you can install ePublisher AutoMap on a Stationery designer or writer computer where ePublisher Express is already installed.

The following figure shows a sample ePublisher configuration.



Requirements

This section lists requirements for ePublisher components and input and output formats.

ePublisher Express, ePublisher Designer, and ePublisher AutoMap Requirements

The following table lists the minimum and recommended system requirements for ePublisher Express, ePublisher Designer, and ePublisher AutoMap.

Note: Memory requirements can vary with the size of the job, including number of files to generate, size of each file, number of images and tables, and more. Generally, performance increases with available memory. The following values provide good performance for an average job.

	Minimum	Recommended
Processor	Intel i3	3.0 GHz Xeon Dual Core
Memory	1 GB RAM	2 GB RAM
Available Disk Space	1 GB available hard disk space	2 GB available hard disk space
Operating System	Microsoft Windows 7 SP1	<ul style="list-style-type: none">• Microsoft Windows 10• ePublisher AutoMap is also supported on Windows Server 2008, 2012 (including R2), 2016.
Additional Software	<ul style="list-style-type: none">• Depending on input requirements• Microsoft .NET Framework 4.6.1 (included with the installer)• Microsoft Visual C++ 2005 Redistributable (included with the installer)	<ul style="list-style-type: none">• Depending on input requirements• Microsoft .NET Framework 4.6.1 (included with the installer)• Microsoft Visual C++ 2005 Redistributable (included with the installer)
Display	800 x 600 display screen resolution	1280 x 1024 display screen resolution (dual monitors supported)

Additional Input Format Requirements

The following table lists the minimum and recommended system requirements for Adobe FrameMaker, Microsoft Word, DITA Open Toolkit and Github-flavored Markdown.

Input Format	Minimum	Recommended
Adobe FrameMaker	FrameMaker 9	FrameMaker 2017
Microsoft Word	Word 2010	Word 2016
DITA Open Toolkit	Oracle JRE or JDK (Java), version 7	Oracle JRE or JDK (Java), version 8
WebWorks Markdown	N/A	N/A

For additional information, please refer to the following page for up-to-date input requirements: <http://wiki.webworks.com/Permalinks/InputRequirements>

Additional Output Format Requirements

You can use ePublisher to produce output in several different formats. This section provides output format requirements for each output format ePublisher supports.

Dynamic HTML

To view Dynamic HTML, users must have a browser that supports HTML 4.0 installed. HTML 4.0 was published in late 1997, and the major browsers, such as Internet Explorer, Firefox, and Safari support HTML 4.0. For more information about the HTML version a browser supports, see the documentation for the browser. If you choose to implement online features that require JavaScript, such as popups, users may also need JavaScript enabled. Most browsers have JavaScript enabled by default. For more information about enabling JavaScript in a browser, see the documentation for the browser.

eBook - ePub 2.0

To generate output in this format, there are no external tools required. However, you will need a compatible ePub reader in order to view the generated output. For development purposes, it is common practice to use an ePub reader on your computer desktop, for example, you can use Adobe Digital Editions available at: [http://www.adobe.com/products/digitaleditions/Microsoft Windows](http://www.adobe.com/products/digitaleditions/Microsoft_Windows).

Eclipse Help

To generate Eclipse Help, you must have the Java2 Platform SDK version 1.2.2 or later installed. You can download the Java2 Platform SDK for free from the Sun Microsystems Web site at <http://java.sun.com/javase/index.jsp>.

If you are using ePublisher to generate Eclipse Help, ePublisher includes a viewer that you can use on the computer where you installed ePublisher to view the Eclipse Help you generated using ePublisher.

To view Eclipse Help when you include Eclipse Help with an application, users must have the Eclipse integrated development environment (IDE) installed. Typically, application developers configure their applications to install the Eclipse IDE with the Eclipse Help content to ensure users can view the Eclipse Help while using the application. To view Eclipse Help, users must also have Microsoft Internet Explorer 6.0 or later or a Mozilla-based browser 1.7 or later installed.

Microsoft HTML Help 1.x

To generate Microsoft HTML Help, you must have Microsoft HTML Help Workshop 1.x installed. If you do not have Microsoft HTML Help Workshop installed, ePublisher will ask you if you want to install Microsoft HTML Help Workshop during the ePublisher installation process. You can also download the Microsoft HTML Help Workshop for free from the Microsoft Developer Network Web site at <http://msdn.microsoft.com/en-us/library/ms669985.aspx>.

To view Microsoft HTML Help, users must have the Microsoft HTML Help viewer installed. The Microsoft HTML Help viewer is installed with most Windows operating systems in use today. Users must also have Internet Explorer 4.0 or later installed. Microsoft HTML Help does not require that users use Internet Explorer as their default browser. Microsoft

Note: Due to the legacy nature of this help run time, if you are generating your help from a networked location, you must map your help drive to a mapped letter such as z:\. UNC drives such as \\server.example.com\directory will not work as output locations for this help format. For more information on this issue, please refer to Microsoft's Support website and search for your version of Windows.

Oracle Help

To generate Oracle Help, you must have the Java2 Platform SDK version 1.2.2 or later installed on your computer. You can download the Java2 Platform SDK for free from the Sun Microsystems Web site at <http://java.sun.com/javase/index.jsp>. The Java 2 Platform is also known as the Java Platform, Standard Edition (Java SE).

To view Oracle Help, users must have the Java Runtime Environment (JRE) installed on their computer. Typically application developers configure their applications to install the JRE with the Oracle Help content to ensure users can view the Oracle Help while using the application. Oracle Help components must be installed and viewed on the local computer.

PDF

Most modern browsers such as Chrome and Microsoft Edge have the ability to read PDF files by default, though the user may also install Adobe Reader if a desktop application is needed. You can download Adobe Reader for free from the Adobe Web Site at http://www.adobe.com/products/acrobat/readstep2_allversions.html.

Customers may encounter issues with font embedding and Windows 7. Please refer to the following resources to address this issue:

<http://wiki.webworks.com/Permalinks/Solutions/Output/PDF/ProblemsInWindows7>

<http://wiki.webworks.com/Permalinks/Solutions/Output/PDF/IssuesWithTrueTypeFonts>

PDF -XSL-FO

To generate PDF - XSL-FO files, you must have the Java2 Platform SDK version 1.2.2 or later installed. You can download the Java2 Platform SDK for free from the Sun Microsystems Web site at <http://java.sun.com/javase/index.jsp>.

Sun JavaHelp 2.0

To generate Sun JavaHelp 2.0, you must have the Java2 Platform SDK version 1.2.2 or later installed on your computer. You can download the Java2 Platform SDK for free from the Sun Microsystems Web site at <http://java.sun.com/javase/index.jsp>. The Java 2 Platform is also known as the Java Platform, Standard Edition (Java SE).

To view Sun JavaHelp, users must have the Java Runtime Environment (JRE) installed on their computer. Typically, application developers configure their applications to install the JRE with the Sun JavaHelp content to ensure users can view the Sun JavaHelp while using the application. Sun JavaHelp components must be installed and viewed on the local computer.

WebWorks Help 5.0

To view WebWorks Help, users must have JavaScript enabled in the browser. If JavaScript is not enabled, then the help system does not display in its entirety. For more information about determining whether JavaScript is enabled in your browser, see your browser options. For more information about enabling JavaScript in Internet Explorer, refer to “Enabling JavaScript”.

WebWorks Help has been tested on the following platforms:

- Internet Explorer
- Microsoft Edge
- Mozilla Firefox
- Netscape
- Safari
- Google Chrome
- Opera

Please refer to the WebWorks wiki for an up-to-date list of supported browsers: <http://wiki.webworks.com/Permalinks/BrowserSupport>

WebWorks Reverb

WebWorks Reverb can now be viewed directly from your computer's file system or from a running web server. This means that you can use this format to deliver online help as part of a non-networked help system. However, in order to provide Reverb's social-media capabilities (i.e. commenting, likes) to your end-users you must deploy the output to a system that is running a web server. Then your end-users must access the content via an **http** or **https** url. If you do not have a web server, you can configure IIS on Windows or any other available web server software. For more information on IIS, consult the following resource.

When you are using ePublisher to generate WebWorks Reverb, ePublisher includes a viewer that you can use on the computer where you installed ePublisher to view the WebWorks Reverb output you generated using ePublisher.

Configuring web server for Reverb

If deploying Reverb output to a public web server that has X-FRAME-OPTIONS configured, make sure that for HTML pages this header is not configured as follows:

```
X-Frame-Options: DENY
```

Instead it should be configured as:

```
X-Frame-Options: SAMEORIGIN
```

Reverb browser requirements

Reverb requires the following browser WebWorks Reverb has been tested on the following platforms:

- Internet Explorer

Note: Reverb 2.0 requires version 11.

- Microsoft Edge
- Mozilla Firefox
- Safari
- Google Chrome

Please refer to the WebWorks wiki for an up-to-date list of supported browsers: <http://wiki.webworks.com/Permalinks/BrowserSupport>

Wiki - Confluence

To view Confluence Wiki output, you must deploy the output to a server computer where Confluence 2.10.2 or later is installed. You may find it helpful to set up a staging Confluence server computer if you generate Confluence output. A staging Confluence computer allows you to view your generated Wiki output before deploying your generated Wiki output to a production Confluence computer.

Before you deploy Confluence output, ensure the following APIs and plug-ins are installed on the Confluence Wiki:

- Confluence Remote API

This API is enabled by default when Confluence is installed. The Confluence Remote API works in conjunction with the Confluence XML RPC API.

- Confluence XML RPC API

This API should be enabled, and this API works in conjunction with the Confluence Remote API.

- Adaptivist Content Formatting Macros plug-in

The Adaptivist Content Formatting Macros plug-in is available at <http://confluence.atlassian.com/display/CONFEXT/Content+Formatting+Macros>. ePublisher uses this plug-in to control indents and any kind of complex table formatting such as row and column spans. This plug-in also provides support for .css classes.

- ***Ensure that the Confluence Compatibility Macros are disabled.*** The Confluence Compatibility Macros define tables. ***If the Confluence Compatibility Macros and the Adaptivist Content Formatting Macros are enabled at the same time***, table macro definition conflicts may result.
- The Confluence WYSIWYG editor does not preserve escaped characters if those character escapes are specified as HTML entities, such as `.`.

Note: **Escape characters** indicate that a character sequence is not formatting instructions for the Wiki. For example, on some Wikis, an asterisk (*) character followed by a space at the beginning of a line indicates a bullet. When the Wiki reads this markup, it converts the asterisk character and the space into a bullet. However, if you need to have an asterisk character followed by a space at the beginning of a line on your Wiki page, and not a bullet, you can use an escape character, such as `'\'` to display the content correctly.

When ePublisher generates output for Wikis, it automatically and appropriately escapes characters in your source documents so your content displays correctly in the generated output. However, if you deploy content with escape characters to a Confluence Wiki server and users then edit the content using the Confluence WYSIWYG editor, the WYSIWYG editor removes any escape characters used on the page to display content. To avoid this issue, disable the Confluence WYSIWYG editor on the Wiki server where you deploy your Confluence Wiki output.

- When you deploy content to a Confluence Wiki, pages are deployed based on the table of contents hierarchy. You can see this hierarchy in the Confluence Browse view.

For more information about Confluence, including installing and configuring Confluence, see the Confluence web site at www.atlassian.com/software/confluence. For more information about deploying Confluence Wiki output using ePublisher, see “Deploying Output”.

Wiki - MediaWiki

To view MediaWiki output, you must deploy the output to a computer where MediaWiki 1.11 or later is installed. You may find it helpful to set up a staging MediaWiki server computer if you generate MediaWiki output. A staging MediaWiki computer allows you to view your generated Wiki output before deploying your generated Wiki output to a production MediaWiki computer.

Before you deploy MediaWiki output, review the following settings on the MediaWiki:

- The user account you specify for ePublisher to use when deploying output to a MediaWiki requires Write access to the Wiki and the ability to remotely push content to the Wiki.
- ePublisher uses the MediaWiki Write API when deploying output to MediaWiki. Using the MediaWiki Write API ensures that any surge protection limits configured on the Wiki do not block ePublisher from deploying output to the Wiki. Ensure that the MediaWiki Write API is enabled before deploying output to MediaWiki. The Write API is disabled by default on MediaWiki.
- If you try to use ePublisher to deploy large page files, such as 2 MB or 3 MB page files, to your MediaWiki server, MediaWiki will display the following error if you do not have enough PHP memory allocated in MediaWiki:

```
Fatal error: Allowed memory size of nnnnnnn bytes exhausted (tried to allocate nnnnnnnn bytes)
```

Ensure you have enough PHP memory allocated in MediaWiki if you plan to use ePublisher to deploy large page files to your MediaWiki server. You can raise the PHP memory limit in the `php.ini` and `LocalSettings.php` files. For more information, see http://www.mediawiki.org/wiki/Manual:Errors_and_Symptoms.

For more information about MediaWiki, including installing and configuring MediaWiki, see the MediaWiki Web site at www.mediawiki.org. For more information about deploying MediaWiki output using ePublisher, see “Deploying Output”.

Wiki - MoinMoin

To view MoinMoin Wiki output, you must deploy the output to a computer where MoinMoin 1.6.4 or later is installed. You may find it helpful to set up a staging MoinMoin server computer if you generate MoinMoin output. A staging MoinMoin computer allows you to view your generated Wiki output before deploying your generated Wiki output to a production MoinMoin computer.

Before you deploy MoinMoin Wiki output, review the following settings on the MoinMoin Wiki:

- The user account you specify for ePublisher to use when deploying output to a MoinMoin Wiki requires Write access to the Wiki and the ability to remotely push content to the Wiki. Ensure the user account ePublisher uses when deploying content to the Wiki has the appropriate Write permissions on the Wiki.
- Ensure `xmlrpc` actions are enabled on the MoinMoin Wiki.

Note: MoinMoin does not support the XML-RPC protocol when run as a CGI application.

- Based on the MIME type assigned to the files you include in your generated output, you may need to add content types to the `mimetypes_embed` setting in the `wikiconfig` file on the MoinMoin Wiki. For example, if you use Scalable Vector Graphics (`.svg`) files when generating MoinMoin Wiki output, ensure that you add `image/svg+xml` to the `mimetypes_embed` setting.
- You can configure surge protection limits for requests on a MoinMoin Wiki, including the type of Write requests ePublisher can make when deploying output to the Wiki. Ensure you understand how surge protection is configured on the MoinMoin Wiki before you use ePublisher to deploy content to a MoinMoin Wiki.

Surge protection limits are typically configured to help protect the performance of the Wiki and to ensure the Wiki is available and has appropriate content for its intended users. Surge protection settings protect the Wiki from malicious users, including spammers, who may want to try and push spam or other unwanted content onto Wikis. In MoinMoin, you configure surge protection limits in the `wikiconfig` file. Verify that ePublisher can deploy content to a MoinMoin Wiki using the surge protection limits configured for the MoinMoin Wiki.

For more information about MoinMoin, including installing and configuring MoinMoin, see the MoinMoin Web site at <http://moinmo.in/>. For more information about deploying MoinMoin Wiki output using ePublisher, see “Deploying Output”.

Downloading ePublisher Installers

ePublisher installers are available for download as .exe files on a secure area on the WebWorks Web site. You can obtain ePublisher installers through one of the following methods:

- ***If you are evaluating ePublisher***, the WebWorks customer service team will send you an email that contains a link to the location where you can download the ePublisher installer.
- ***If you are a new ePublisher customer***, the WebWorks customer service team will send you an email that contains a link to the location where you can download the ePublisher installer when you purchase ePublisher.
- ***If you are an existing ePublisher customer with an active maintenance agreement***, the WebWorks customer service team will automatically send you an email that contains a link to the location where you can download the ePublisher installer each time a new version of ePublisher releases. If you have a My Cases login for the WebWorks technical support Web site, you can also obtain the ePublisher installer in the My Cases area when you log in to the WebWorks technical support site.
- ***If you are an existing ePublisher customer without an active maintenance agreement***, contact the WebWorks account management team for more information.

The link you receive to the download location for the ePublisher installer is typically active for only one to two weeks. ePublisher installer download locations are changed often for security reasons. If you need the latest link to an ePublisher download kit, you can request a link by submitting a support request on the WebWorks Web site at <http://www.webworks.com/Support/>. WebWorks technical support will verify that you purchased an ePublisher license for the requested component and then provide a link where you can download the requested installer.

To download an ePublisher installer

1. Click the download link in the email from WebWorks.
2. On the WebWorks download page, click the link for the ePublisher component you want to install.
3. Click **Save**.
4. Browse to a location on your local computer where you want to save the installer, and then click **Save**.
5. Click **Close** when the download completes.
6. Browse to the location on your local computer where you saved the .exe file for the ePublisher component.
7. Run the .exe file.

Microsoft Windows Requirements

ePublisher components require the *Microsoft Visual C++ 2005 SP1 MFC Security Update Redistributable Package (x86)*, which is installed by default as of Windows 10.

Note: The ePublisher Express installer will handle the download and installation of this component if it is not present on the user's machine. If the user cannot connect to the internet while installing, then this component will need to be downloaded and installed manually before proceeding with the ePublisher Express installation. This component can be found for free at: https://download.microsoft.com/download/8/B/4/8B42259F-5D70-43F4-AC2E-4B208FD8D66A/vcredist_x86.EXE.

To see if the *Microsoft Visual C++ 2005 SP1 MFC Security Update Redistributable Package (x86)* is already installed on your computer: Open the **Control Panel**, and select **Add or Remove Programs** and check if the *Microsoft Visual C++ 2005 Redistributable* is listed.

Downloading and Installing the Microsoft .NET 4.6.1 Framework

ePublisher components require the Microsoft .NET 4.6.1 Framework.

Note: The ePublisher Express installer will handle the download and installation of this component if it is not present on the user's machine. If the user cannot connect to the internet while installing, then this component will need to be downloaded and installed manually before proceeding with the ePublisher Express installation. This component can be found for free at: <http://go.microsoft.com/fwlink/?LinkId=671743>. In any case, after the installation of the Microsoft .NET 4.6.1, the user needs to restart the computer before installing ePublisher Express.

To see if the Microsoft .NET 4.6.1 Framework is already installed on your computer: Open the **Control Panel**, and select **Add or Remove Programs** and check if the *Microsoft .NET Framework 4.6.1* is listed.

Enabling JavaScript

To use ePublisher, you must enable JavaScript in Microsoft Internet Explorer.

To enable JavaScript in Internet Explorer

1. Open Microsoft Internet Explorer.
2. On the **Tools** menu, click **Internet Options**.
3. On the Security tab, click **Custom Level**.
4. Under **Active Scripting**, click **Enable**.

Installing ePublisher

This section explains how to install ePublisher components. Read this section before you install ePublisher components.

Installation Order for ePublisher Components

Ensure you install ePublisher components in the correct order. Review the following installation options before you install ePublisher components:

- ***If you want to generate output using Stationery created by a Stationery designer***, install only ePublisher Express on the computer.
- ***If you want to design Stationery and generate output***, install ePublisher Express and ePublisher Designer on the computer.

Note: ePublisher Designer requires ePublisher Express. Ensure you install ePublisher Express on the computer before you install ePublisher Designer.

- ***If you want to schedule and automate output generation***, install ePublisher Express and ePublisher AutoMap on the computer where you want to use ePublisher AutoMap.

ePublisher AutoMap requires ePublisher Express. Ensure you install ePublisher Express on the computer before you install ePublisher AutoMap. You can install ePublisher AutoMap using one of the following configurations:

- On its own separate computer where ePublisher Express is already installed
- On a Stationery design computer where ePublisher Express and ePublisher Designer are already installed
- On a writer computer where ePublisher Express is already installed

Installing ePublisher Components

This section provides instructions for installing ePublisher components, including ePublisher Express, ePublisher Designer, and ePublisher AutoMap.

Note: You must install ePublisher Express first. Then, you can install ePublisher Designer and ePublisher AutoMap. Both ePublisher Designer and ePublisher AutoMap require ePublisher Express.

If you are upgrading from a previous version of ePublisher, review the upgrade instructions. For more information, see “Upgrading from Previous Versions”.

To install ePublisher components

1. Log on as a user using an account that is a member of the Administrators group on the local computer.
2. Close all instances of Microsoft Office applications running on the local computer, including instances of Microsoft Word and Microsoft Outlook. Close all instances of Adobe FrameMaker running on the computer.
3. Run the `WebWorks ePublisher <Product><Build_Number>.exe` file for the ePublisher component you want to install.
4. Review the welcome message, and then click **Next**.
5. Review the license agreement. If you agree to the terms of the agreement, click **I Agree**.
6. Select the application shortcuts you want to create, and then click **Next**.
7. Specify the location of the installation directory, and then click **Next**. The default installation directory is `c:\Program Files (x86)\WebWorks`.
8. Click **Next** to confirm your selections and to begin installing the ePublisher component.
9. In the WebWorks Licensing Info window, complete the following steps:
 - a. Enter your Contract ID. If you previously installed ePublisher on the computer using a valid Contract ID, ePublisher will automatically detect the Contract ID and display your Contract ID information. For more information about contract IDs and obtaining a Contract ID, see “Working with Contract IDs” “Obtaining Contract IDs”.
 - b. Enter your email address. If you have an email address that you use as your WebWorks support login, enter that email address.
 - c. Enter the name of your computer.
 - d. Click **Confirm**.
10. *If the installer displays the **HTML Help Workshop 1.3 Setup** window*, you can install Microsoft HTML Help Workshop 1.3 as part of your ePublisher installation. Install Microsoft HTML Help Workshop if you plan to generate Microsoft HTML Help output.
 - *If you want to install Microsoft HTML Help Workshop 1.3*, click **Yes**, and then follow the instructions to install Microsoft HTML Help Workshop.
 - *If you do not want to install Microsoft HTML Help Workshop 1.3*, click **No**.
11. Click **Close** when the installation completes. ePublisher also opens a new browser window and displays a page on the www.webworks.com web site when the installation completes.

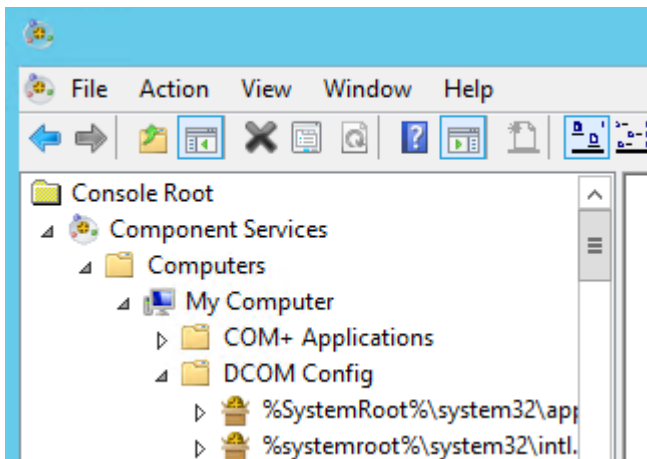
12. Restart your computer to update and register all configuration files.

Configuring AutoMap for Microsoft Word Inputs

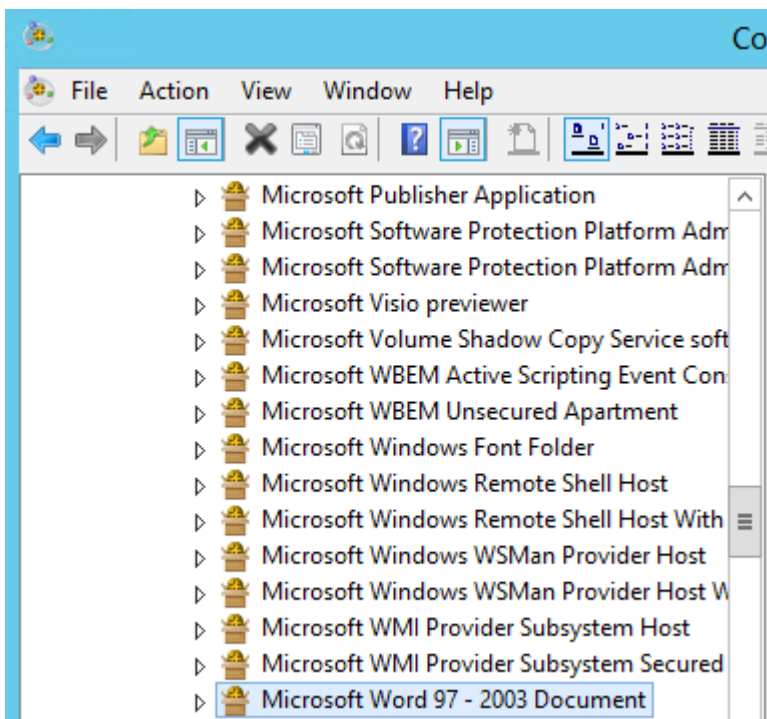
When installing ePublisher AutoMap and if publishing Microsoft Word documents, it may be necessary to configure the DCOM **Login Identify** for the **Microsoft Word 97 - 2003 Document** configuration. Depending on how you or your team will be running ePublisher AutoMap the user account that Microsoft Word will be launched as should be set accordingly.

To configure the DCOM account Identify of Microsoft Word

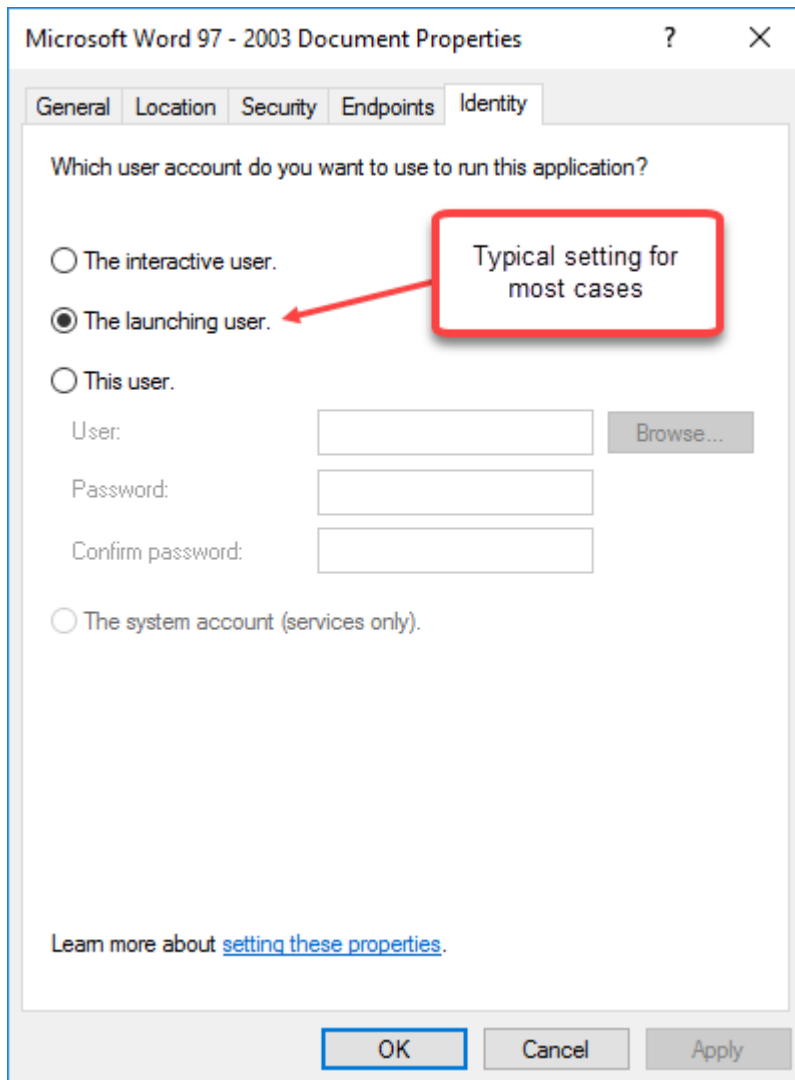
1. *If working with a 32-bit version of Microsoft Word*, from the **Start** menu, type: `mmc comexp.msc /32`
2. *If working with a 64-bit version of Microsoft Word*, from the **Start** menu, type: `mmc comexp.msc`
3. In the Component Services dialog, select **Component Services > Computers > My Computer > DCOM Config**.



4. Underneath **DCOM Config**, right-click **Microsoft Word 97 - 2003 Document** and select **Properties**.



5. In the **Microsoft Word 97 - 2003 Document Properties** dialog, select the **Identity** tab, then specify the user account to use to run Microsoft Word, which may or may not be a specific user depending on how you are planning to use ePublisher AutoMap.



Understanding Installed Sample Projects and Stationery

ePublisher Express and ePublisher Designer install sample projects and Stationery. You can use these sample projects and Stationery to see some examples of how you can use ePublisher to generate output. For more information about using these sample projects and Stationery to generate output, see the *ePublisher Evaluation Guide* or *ePublisher Writer Guide*.

Working with Contract IDs

ePublisher no longer requires you to manually enter license keys. ePublisher now uses Contract IDs to enable product functionality, which simplifies the ePublisher licensing process.

A Contract ID is a unique identifier that identifies the number of users and type of functionality enabled for your ePublisher installation. WebWorks generates an appropriate Contract ID for your ePublisher installation when you purchase ePublisher or request an evaluation copy of ePublisher. A Contract ID enables functionality based on the items and time frame specified in the purchase contract between your company and WebWorks.

If you have a valid contract ID for one version of the ePublisher product, when a new version of ePublisher releases, you can continue to use your same Contract ID when you upgrade to the new version of the product. You can also continue to use your same Contract ID if you have to uninstall and then re-install a version of ePublisher.

ePublisher licensing is flexible, and the WebWorks team can work with you ensure that you have the licensing that is right for you. Contact WebWorks Sales at sales@webworks.com or WebWorks Customer Service at customerservice@webworks.com to discuss any special licensing needs you may have.

Viewing Licensing and Contract ID Information

You can view licensing information in the License Information window in ePublisher. ePublisher uses adapter license keys, or activation codes, to enable ePublisher functionality. Adapter licensing information is specified in your Contract ID. ePublisher uses an Internet connection to connect to the ePublisher licensing server and periodically retrieve and update adapter activation codes as needed based on your Contract ID.

Note: If you need to install ePublisher in an environment without Internet connectivity, WebWorks can provide Contract IDs that support this environment. For more information, see “Managing Licensing in Environments without Internet Connectivity”.

ePublisher licenses, or activation codes, do not display in the ePublisher user interface, but you can view the adapters for which you are licensed and your Contract ID number in the ePublisher user interface.

To view ePublisher licensing and Contract ID information

1. On the **Help** menu, click **License Keys**. ePublisher displays the input formats for which the component is licensed in the License Information window.
2. *If you want to view your Contract ID number*, click **Register**.

Obtaining Contract IDs

ePublisher now uses Contract IDs instead of license keys to enable ePublisher functionality.

If you are evaluating ePublisher, the WebWorks customer service team will send you an email that contains a Contract ID you can use when you install an evaluation copy of ePublisher. If you have not received an evaluation Contract ID or are having problems with your evaluation license, send an email to customerservice@webworks.com.

If you are a new ePublisher customer, the WebWorks customer service team will send you an email that contains your Contract ID when you purchase ePublisher. If you have not received a Contract ID or are having problems with your licensing, send an email to customerservice@webworks.com.

If you are an existing ePublisher customer with an active maintenance agreement, the WebWorks customer service team will automatically send you an email that contains a link to the location where you can download the ePublisher installer. ePublisher will automatically detect and use your existing Contract ID each time you install a new version of the ePublisher product. If you have not received a Contract ID or are having problems with licensing, send an email to customerservice@webworks.com or submit a support request.

If you are an existing ePublisher customer without an active maintenance agreement, contact the WebWorks account management team for more information about obtaining your Contract ID by sending an email to sales@webworks.com.

For more information about Contract IDs, see “Working with Contract IDs” and “Entering Contract IDs”.

Entering Contract IDs

ePublisher now uses Contract IDs instead of license keys to enable ePublisher functionality. You must enter your Contract ID, email address, and computer name before you can use ePublisher components. The Contract ID enables the ePublisher product components and ePublisher input formats for which you are licensed. For more information about Contract IDs, see “Working with Contract IDs”.

To enter a Contract ID

1. On the **Help** menu, click **License Keys**. ePublisher displays the input formats for which the component is licensed in the License Information window.
2. Click **Register**.
3. In the **Contract** field, enter your Contract ID.
4. In the **Email** field, enter your email address. If you have an email address that you use as your WebWorks support login, enter that email address.
5. In the **Computer** field, enter the name of the computer where you are installing ePublisher.
6. Click **Confirm**.

Managing Licensing in Environments without Internet Connectivity

ePublisher uses an Internet connection to connect to the ePublisher licensing server and retrieve or update adapter activation codes as needed based on your Contract ID. If you need to install ePublisher in a restricted environment where ePublisher computers do not have Internet access, contact WebWorks Sales at sales@webworks.com or WebWorks Customer Service at customerservice@webworks.com to request a non-network Contract ID. The ePublisher licensing model is flexible, and WebWorks can work with you to provide non-network Contract IDs or other licensing solutions appropriate for your environment.

Updating Licensing

ePublisher automatically contacts the ePublisher licensing server as needed to obtain updated activation codes. ePublisher communicates with the ePublisher licensing server using an Internet connection. ePublisher obtains updated activation codes as appropriate based on the licensing specified in your Contract ID.

Note: If your ePublisher is installed in an environment without Internet connectivity, WebWorks can provide Contract IDs that support this environment. For more information, see “Managing Licensing in Environments without Internet Connectivity”

Typically, you will not need to request updated activation codes, as ePublisher obtains updated codes for your automatically. However, you can manually request updated activations codes in the ePublisher interface. For example, you may want to manually request updated activation codes if you know that the computer where you installed ePublisher will not have Internet access for a long period of time. When you request updated activations codes in the ePublisher interface, ePublisher immediately establishes an Internet connection to the ePublisher licensing server and automatically obtains updated activation codes for the ePublisher adapters for which you are licensed.

To update ePublisher licensing

1. On the **Help** menu, click **License Keys**. ePublisher displays the input formats for which the component is licensed in the License Information window.
2. Click **Refresh keys**. ePublisher retrieves updated activation codes from the ePublisher licensing server.

Deactivating Licensing

You can deactivate ePublisher licensing in the ePublisher user interface. Deactivating licensing for the current ePublisher installation allows you to install ePublisher on a different computer without affecting the number of available seats allowed by your contract.

Note: The terms of the ePublisher end-user license agreement (EULA) allow you to install ePublisher Express or ePublisher Designer on one office computer and on one home or travelling computer for each assigned ePublisher user seat. ePublisher AutoMap licensing terms can vary based on whether ePublisher AutoMap was purchased on a per writer or per server basis, or in conjunction with a Content Management System (CMS).

To deactivate ePublisher licensing:

1. On the **Help** menu, click **License Keys**.
2. Click **Unregister**.

Upgrading from Previous Versions

In most cases, upgrading from a previous version to a new version of ePublisher can be accomplished in just a few steps. This section explains how to prepare for an upgrade, how to upgrade a typical ePublisher installation, and how to upgrade an ePublisher implementation with advanced customizations.

Updating ePublisher installation

If you are installing an ePublisher component that is of version 2018.1 or higher and your currently installed component is version 2017.1 or higher, then your component will automatically be uninstalled for you when you choose the option to update. Otherwise, before installing a new version of an ePublisher component, you must uninstall any previous versions of the component. Uninstalling an ePublisher component removes the installation folder and registry entries for the component from the computer.

To update/repair an ePublisher component with an ePublisher executable installer

1. Close all ePublisher user interfaces.
2. Close all instances of Microsoft Office applications running on the local computer, including instances of Microsoft Word and Microsoft Outlook. Close all instances of Adobe FrameMaker running on the computer.
3. Double click the executable installer.
4. Select whether you want to Update/Repair, and then click **Next**.
5. Follow the instructions in the consecutive pages.

Preparing existing projects for ePublisher Upgrade

To ensure smooth migration of existing projects and stationeries perform the following steps:

- Save your Stationery, Stationery design projects, and any projects you currently use to generate output to a secure location. **Stationery** defines the appearance and functionality of all the output formats you need. **Stationery design projects** are the ePublisher Designer projects used to create Stationery. For more information about Stationery and Stationery design projects, see “Understanding Stationery” and “Creating a Stationery Design Project”.
- *If you implemented overrides when designing Stationery*, ensure you save the files in the `Formats` folder, any override files currently in use, and a copy of the original files from which the overrides were created to the secure location. Examples of overrides include the following items:
 - Modifications to the `Page.asp` file
 - Custom `.css` files
 - Modifications to image files
 - Any advanced overrides such as modifications to `.xsl` or `.fti` files or files in the `Formats` folder.

When the Stationery designer creates and saves Stationery, ePublisher creates the following folders:

- `StationeryName\Formats\OutputFormat`
- `StationeryName\Formats\OutputFormat.base`

where `StationeryName` is the name the Stationery designer specified for the Stationery, and `OutputFormat` is the type of output format the Stationery Designer specified for a target in the Stationery.

The `StationeryName\Formats\OutputFormat` folder contains any customizations or overrides the Stationery designer specified when designing the Stationery. ePublisher Express synchronizes with the files in the `OutputFormat` folder and uses the information about customizations and overrides contained in files in the `OutputFormat` folder to generate output.

Note: The Stationery may have one or more `OutputFormat` folders, based on the settings the Stationery designer specified.

The `StationeryName\Formats\OutputFormat.base` folder contains copies of all the files located in the `\Program Files\WebWorks\ePublisher\release_number\Formats\OutputFormat` folder. These files define the default output format and transforms and are installed by default when you install ePublisher.

Stationery designers can do a compare, or **diff**, between the files located in these folders to quickly see any customizations or overrides specified for the Stationery. Stationery designers can use this information to help them reapply customizations and overrides as needed when designing a newer version of the Stationery in ePublisher Designer.

For more information about overrides, see “Understanding Stationery, Projects, and Overrides”.

Upgrading Typical ePublisher Implementations

After you save your existing Stationery design projects, Stationery, any projects you currently use to generate output, and any copies of override files to a secure location, perform the following steps:

- On the Stationery designer computer, uninstall all existing versions of ePublisher components, such as ePublisher Express, ePublisher Designer, and ePublisher AutoMap. The **Stationery design computer** is the computer the Stationery designer uses to create and update Stationery. ePublisher Express and ePublisher Designer are installed on the Stationery design computer. Based on your configuration, ePublisher AutoMap may also be installed on the Stationery design computer.
 - Install the new version of ePublisher Express and ePublisher Designer on the Stationery designer computer. Also install the new version of ePublisher AutoMap if you run ePublisher AutoMap on the Stationery designer computer.
 - Open your existing Stationery design projects using the new version of ePublisher Designer.
 - Generate output and verify that your output generates as expected. Make any adjustments as needed.
 - *If you have implemented typical overrides in a Stationery design project*, such as overrides to `Page.asp` files, custom `.css` files, or `image` files, you can continue to use your overrides to these files, and the new version of ePublisher will recognize and use these existing modifications when generating output.
 - *If you have implemented advanced overrides*, such as overrides to `.xsl` or `.fti` files, or overrides to files in the `Formats` folder, update these files in your new ePublisher installation to include your advanced overrides. For more information, see “Upgrading Implementations with Advanced Customizations”.
- Note:** When ePublisher Designer detects overrides, by default it will not update to the latest version of the format. This means that no modifications will be necessary in order to continue using your Stationery. However, in this default mode, you will not get any of the format improvements built into the latest release. If you want these improvements, then you will have to configure the **Project Settings** to use the latest version of ePublisher’s formats.
- Create new Stationery for each Stationery design project.
 - Deploy the updated Stationery to an appropriate location.
 - On each writer computer, update ePublisher Express installation.
 - The next time writers generate output, they open their existing projects using the new version of ePublisher Express. Writers can choose to synchronize their projects immediately to obtain the latest Stationery and then generate output, or writers can continue using their existing Stationery until they are ready to move to the latest version of the Stationery.

Upgrading Implementations with Advanced Customizations

If you have implemented advanced overrides in the Stationery design, such as overrides to `.xsl` or `.fti` files, or overrides to files in the `Formats` folder, ensure you save a copy of the following items to a secure location before uninstalling a previous version of ePublisher and installing a new version:

- Overrides currently used in the Stationery design project
- A copy of the original files from which the overrides were created

If you want to continue to use your advanced customizations with the new version of ePublisher, first uninstall your previous ePublisher version and then install a new ePublisher version. Then identify and include your overrides in the new versions of the ePublisher files as appropriate by performing a three-way merge of the following items:

- A copy of the existing override file used in the Stationery design project, located in the `StationeryName\Formats\OutputFormat` folder, where `StationeryName` is the name the Stationery designer specified for the Stationery, and `OutputFormat` is the type of output format the Stationery Designer specified for a target in the Stationery.
- A copy of the original file from which the override was created, available in the `StationeryName\Formats\OutputFormat.base` folder, where `StationeryName` is the name the Stationery designer specified for the Stationery, and `OutputFormat` is the type of output format the Stationery Designer specified for a target in the Stationery.
- A copy of the new file from the new version of ePublisher

Performing a three-way merge allows you to identify the code you changed when you created the override, and also allows you to quickly and easily create the override again in the new ePublisher files. You may find tools such as Araxis Merge Pro, available at <http://www.araxis.com/merge>, or KDiff3, available at <http://kdiff3.sourceforge.net>, helpful as you compare and merge override files.

After you perform your three-way merge and update the files you want to override in the new version of ePublisher with the overrides you specified in the previous version, test your overrides by generating output using the new version of ePublisher Designer and the Stationery design project to confirm your output generates appropriately. After you verify the output generated correctly using your advanced customizations, you can create new Stationery using the Stationery design project and then deploy the updated Stationery that includes your advanced customizations to writers to use to generate output.

Upgrading Advanced Customizations of WebWorks Reverb Skin

The WebWorks Reverb output format is highly customizable and chances are you may have made advanced customizations to this format. If you are upgrading from a prior release, then you will want to understand what files are most likely to be customized and how this is affected when you change the Skin **Target Setting**.

If you have or plan to customize any of the WebWorks Reverb skins, then most likely you will have to modify one or more of the following files.

Table 5: Commonly Customized Reverb Files

Filename	The display area or items affected by this file
webworks.css	Content panel styling only. Includes the styling of the MiniTOC, RelatedTopics, Social Buttons.
skin.css	Styling of TOC, Index, Toolbar, and Breadcrumbs. All icons used in the <code>skin.png</code> sprite file are managed here. Styling of content that appears above the Toolbar, such as the company information.
search.css	Styling of the search results page.
skin.png (derived from Skin.Fireworks.png)	PNG file with alpha channel that stores all of the Reverb icons.
connect.asp	Used to manage the button placement in the toolbar. Also manages the TOC/Index/Search panel title for the <i>Corporate</i> skin.
connect.css	Manages basic structure of the entry-point file generated from the <code>connect.asp</code> template file.

When working with alternate skins, you need to be aware of which files are most likely affected as a result of changing the skin type. If you have Advanced Customizations in any of these files, then you need to re-examine the *diffs* of these files after you switch the skin type. Most likely you will see significant changes. Here are some basic steps you can follow to make sure you translate those changes to the new skin properly.

Basic steps for setting an alternate skin type when Advanced customizations are present

1. Check your **Advanced Customizations** for files listed in “Commonly Customized Reverb Files”.
2. Make sure any of these commonly customized files are implemented as **Target Overrides** as opposed to **Format Overrides**. Setting an alternate skin type will create an implicit target override that will have priority over any format overrides of the same name.
3. Before changing the skin type you will need to record any existing file differences. On the **Advanced** menu click **Manage Target Customizations**. Now use the procedure discussed in “Managing Format/Target Overrides” to record these file differences. These file differences will be used later after the skin type has been changed.
4. On the **Target** menu, click **Target Settings**.
5. In the **WebWorks Reverb** category, select the right column of the **Skin** entry to display the file picker button.
6. Click the file picker button to bring up an **Open** file dialog which will display a list of skin plugin files. Each skin plugin file is identifiable by a `.weplugin` extension.
7. Browse to the plugin file that you wish to use and double-click it to set the skin to that value.

8. At this point, you need to consider either removing your existing customizations and then re-implementing them using the information from your previously recorded file differences. Or managing the differences directly by comparing the differences using the procedure discussed in “Managing Format/Target Overrides”. Either method will work.

Uninstalling ePublisher

Uninstalling an ePublisher component removes the installation folder and registry entries for the component from the computer.

If ePublisher installed the WebWorks Transit menu for Microsoft Word on the computer, ePublisher removes the WebWorks Transit menu and WebWorks Transit registry entries when you uninstall the last ePublisher component on the computer.

To uninstall an ePublisher component with an ePublisher executable installer

1. Close all ePublisher user interfaces.
2. Close all instances of Microsoft Office applications running on the local computer, including instances of Microsoft Word and Microsoft Outlook. Close all instances of Adobe FrameMaker running on the computer.
3. Double click the executable installer.
4. Select the Uninstall option, and then click **Next**.
5. Follow the instructions in the consecutive pages.

To uninstall an ePublisher component using Windows Control Panel

1. Close all ePublisher user interfaces.
2. Close all instances of Microsoft Office applications running on the local computer, including instances of Microsoft Word and Microsoft Outlook. Close all instances of Adobe FrameMaker running on the computer.
3. Open Control Panel.
4. Open Add or Remove Programs.
5. Select the ePublisher component you want to uninstall.
6. Click **Remove**.
7. Click **Yes** to confirm you want to remove the ePublisher component from your computer. ePublisher removes the selected ePublisher component.

Troubleshooting Installation, License Keys, and Uninstallation

This section helps you troubleshoot issues related to the following ePublisher issues:

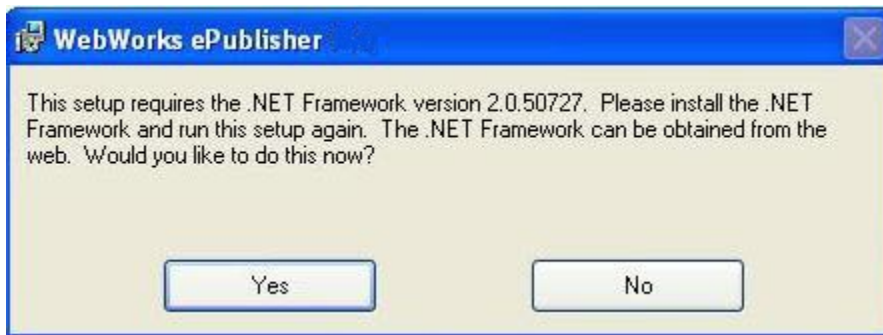
- Installing ePublisher. For more information, see “Problems Installing ePublisher”.
- Obtaining, adding, and removing Contract IDs and working with licensing. For more information, see “Problems with Contract IDs and Licensing”.
- Uninstalling ePublisher. For more information, see “Problems Uninstalling ePublisher”.

Problems Installing ePublisher

This section helps you troubleshoot issues related to installing ePublisher.

Error: Microsoft .NET Framework 2.0 Not Installed

If you are installing an ePublisher component and you do not have the Microsoft .NET Framework installed, ePublisher displays the following error message.

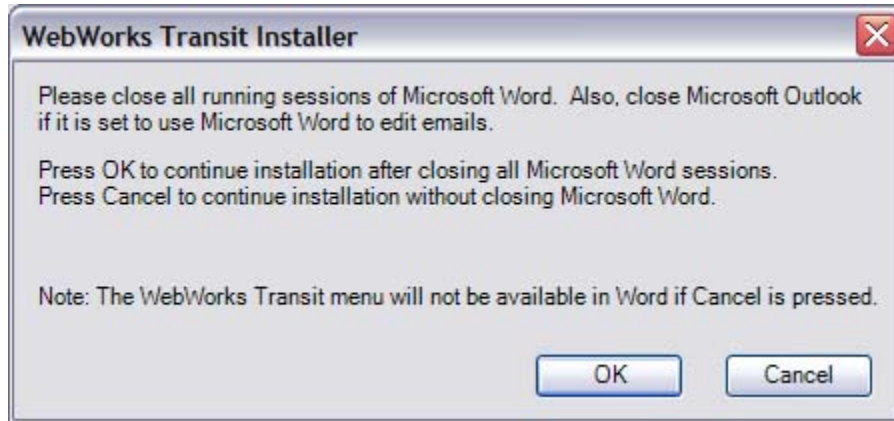


To resolve this issue

1. Click **Yes**. ePublisher redirects you to the Microsoft .NET Framework Web site.
2. Download and install the Microsoft .NET Framework Redistributable Package. You do not need to download and install the Microsoft .NET Framework Software Development Kit.

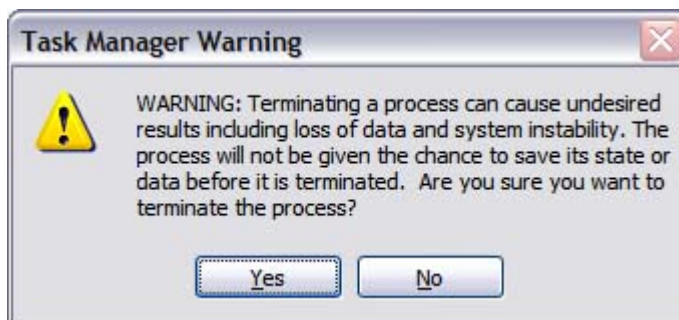
Error: Please Close all Running Sessions of Microsoft Word

If you have any Microsoft Office processes running when installing ePublisher, including instances of Microsoft Word and Microsoft Outlook, ePublisher displays the following error message.



To resolve this issue

1. Close any running instances of Microsoft Word.
2. Close Microsoft Outlook.
3. Open Task Manager.
4. Click on the **Processes** tab.
5. Search for `WINWORD.EXE`. You can click on the **Image Name** column to sort the processes alphabetically.
6. *If there is a **WINWORD.EXE** process running*, complete the following steps:
 - a. Select `WINWORD.EXE`.
 - b. Click **End Process** to close all running Word processes. Task Manager displays the following warning.



- c. Click **Yes**.
 - d. Close Task Manager, and then proceed with your ePublisher installation.
7. *If there are no **WINWORD.EXE** processes running*, proceed with your ePublisher installation.

Problems with Contract IDs and Licensing

This section helps you troubleshoot issues related to obtaining, adding, and removing Contract IDs. For more information about Contract IDs, see “Working with Contract IDs”.

No Contract ID Received

After you purchase ePublisher components, your WebWorks customer service team will e-mail your Contract ID that enables licensing for the products your purchased. If you have not received a Contract ID, send an email to sales@webworks.com. For more information about Contract IDs, see “Working with Contract IDs”.

Error: No Valid License Key Found

You must enter a Contract ID before you can generate output. If you have not entered your Contract ID information, ePublisher displays an error stating that no valid license key was found to enable support for your content authoring tool.

If you have entered a Contract ID but still receive this error, verify you entered your Contract ID information correctly. For more information about Contract IDs, see “Working with Contract IDs”.

Other Contract ID and Licensing Problems

If you have received your Contract ID and entered your Contract ID into ePublisher but you are having problems with licensing, send an email to customerservice@webworks.com. For more information about Contract IDs, see “Working with Contract IDs”.

Note: ePublisher licensing is flexible, and the WebWorks team can work with you ensure that you have the licensing that is right for you.

Problems Uninstalling ePublisher

This section helps you troubleshoot issues related to uninstalling ePublisher.

Error: You Must Remove the Previous Version of ePublisher

If you try to install a new version of an ePublisher component before uninstalling the previous version of an ePublisher component, ePublisher displays an error message telling you need to remove the previous version of the ePublisher component before you can install the new version of the component. ePublisher should no longer display this error once you uninstall the previous version of the component.

If after you uninstall an ePublisher component, you still receive an error when you try to install a new version of the component, confirm that the component is no longer listed in the Add or Remove Programs list in Control Panel. If the component is no longer listed but you still receive the error, there may still be some registry keys from the previous installation on the computer that are preventing you from installing a new version of the ePublisher component.

To resolve this issue

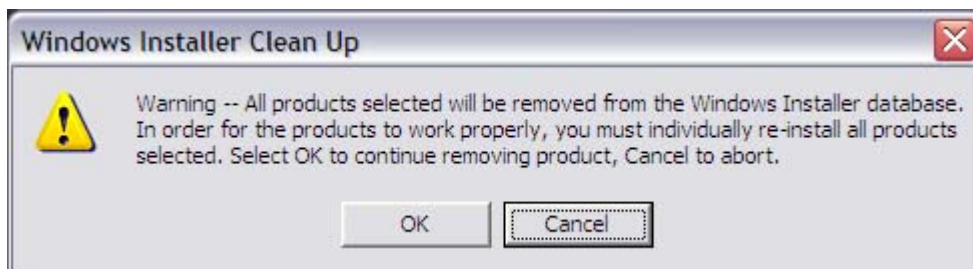
1. Log on to the WebWorks technical support site at http://www.webworks.com/Support/My_Cases/.
Note: You must have an active maintenance agreement in order to log on to the WebWorks technical support site.
2. On the **Find Solution** tab, in the **Search for** field, type `Problems installing or upgrading ePublisher component`, and then click **Find Solution**.
3. Open the Knowledge Base article.
4. Click the **Remove WebWorks registry file** link.
5. Download the `.zip` file.
6. Unzip the file and then run the `.reg` file.
7. Follow the instructions to remove any problematic keys. No other changes will be made to your computer.

Problems Completely Uninstalling ePublisher

If you have a problem uninstalling an ePublisher component, first ensure that no registry keys from the previous version of the component remain on your computer. For more information, see “Error: You Must Remove the Previous Version of ePublisher”. If after removing all registry key entries from the previous version you still are unable to completely uninstall ePublisher, download and install the Windows Installer Cleanup Utility. The Windows Installer Cleanup Utility allows you to completely remove ePublisher Express from your system.

To completely remove ePublisher using the Windows Installer Cleanup Utility

1. Download and install the Windows Installer Cleanup Utility. The Windows Installer Cleanup Utility is available on the Microsoft Web site at <http://support.microsoft.com/?scid=kb;en-us;290301>.
2. Open the Windows Installer Cleanup Utility.
3. Find and select the ePublisher component from the list of installed products.
4. Click **Remove**.
5. A window displays and alerts you that ePublisher component will be deleted from the Windows Installer database. Click **OK** to remove the ePublisher component.



6. Browse to `Program Files\WebWorks` on your local computer.
7. Verify that the ePublisher component is no longer in the folder. If the folder still exists, delete it.

Other Errors Uninstalling ePublisher

If you have already tried to completely uninstall the ePublisher component using the Windows Installer Cleanup Utility but you are still receiving errors, submit a support request on the WebWorks Web site at http://www.webworks.com/Support/My_Cases/.

3

Selecting Input and Output Formats

ePublisher allows you to use multiple input formats, such as Adobe FrameMaker, Microsoft Word, and DITA files, to generate the output you need. Each input format provides strengths and challenges for authors. To meet the wide variety of product and company needs, ePublisher generates many different output formats from your source documents.

Understanding the ePublisher Process and Workflow

With ePublisher, a **Stationery designer** creates Stationery that identifies the supported output formats and defines the appearance and behavior of the output generated with projects based on that Stationery. A **writer** creates content source documents using one of the supported input formats. The writer then creates a project based on the Stationery, identifies the source documents to include in the project, and uses the project to generate output for each target defined in the project.

When generating output, ePublisher applies the defined conditions, variables, and cross-reference formats to the source documents. ePublisher next exports the source document content to the WebWorks Intermediate Format (WIF). This format provides a standard for ePublisher to process. Then, ePublisher applies the styles and options defined in the project and generates the selected output using XSL transforms that you can customize if needed. For more information, see “Understanding How ePublisher Works”.

Selecting Your Input Formats

In many cases, companies have specific tool or source format requirements, or previous decisions were made that determine the input formats you need to support. ePublisher allows you to mix multiple input formats and combine them into a single deliverable in its generated output. With this powerful solution, you can use the input formats you need, which allows authoring teams to use the tools and formats they require to be most effective. For more information about each of the supported input formats and their strengths, see “Understanding Input Formats”.

Selecting Your Output Formats

ePublisher supports many output formats and you can implement many powerful features in your online content. Some output formats are designed to support specific use cases and environments. In addition, some features are available only in certain output formats. For more information about using various features in a specific output format, see “Features Available in Each Output Format”.

You need to consider your goals, the environment where your online content will be used, and the features you want to use in your content to decide which output format is best:

- To deliver a full-featured online help system with a navigation pane, content pane, full-text search, and other common help features, ePublisher provides several help-related output formats. For more information about selecting the output format to meet your specific requirements, see “Understanding Input Formats”.
- To deliver web-based content, ePublisher provides several web-based output formats. For more information about selecting the output format to meet your specific requirements, see “Selecting Your Web Content Format” on page 49.
- To deliver a file that is easy to download and print, consider generating either the PDF or PDF - XSL-FO output format for this purpose. Both formats will generate a resulting PDF, however the PDF - XSL-FO output format provides a complete capability for customizing and/or controlling the style and layout of the generated PDF.
- To integrate your content with an XML-based solution or a company process that needs XML input, consider generating the XML+XSL output format.

Understanding Input Formats

The following sections summarize the default input formats supported by ePublisher. You can implement many powerful features in your generated online content. Some features are supported only with certain input and output formats. In addition, how you define these online features in your source documents can differ based on your input formats.

Adobe FrameMaker

Adobe FrameMaker provides a comprehensive publishing solution with XML-based structured authoring. You can develop the templates you need to deliver polished technical documentation for large product libraries. FrameMaker allows you to create both structured and unstructured content. You can also create DITA-compliant content. FrameMaker provides a good solution for the following conditions:

- Long source documents
- Many images included in your source documents
- Multiple page layouts used throughout your source documents
- Conditions needed to deliver multiple versions of your source documents
- Multiple files to allow multiple writers to work simultaneously on the content
- Comprehensive format controls, such as keep with previous paragraph

For more information about using FrameMaker and establishing single-sourcing standards with FrameMaker, see “Designing Adobe FrameMaker Formats and Standards”.

Microsoft Word

Microsoft Word is a powerful authoring tool that allows you to quickly create professional content. You can develop templates with the styles and other standard elements you need to deliver polished technical documentation. Microsoft Word provides many automation features to streamline the content development process. The new XML-based formats allow Microsoft Word to integrate content with other Microsoft Office products. Microsoft Word provides a good solution for the following conditions:

- Common authoring tool needed across departments in your company
- Reducing costs for authoring tools is an important consideration
- Source documents are less than 500 pages with some graphics throughout. Please keep in mind that Word itself does not generally do well when editing 100+ page documents.
- Automation, such as toolbars and macros, needed to streamline content creation
- Basic conditions needed to show or hide portions of your source documents
- Rarely need to allow multiple writers to work simultaneously on the same content

For more information about using Microsoft Word and establishing single-sourcing standards with Microsoft Word, see “Designing Microsoft Word Templates and Standards”.

DITA XML Files

DITA (Darwin Information Typing Architecture) is an XML-based format for creating and publishing technical content. DITA leverages the strengths of XML and provides a standard set of element definitions used to create technical content. These definitions include three topic types based on the standard topic definition:

- Task
- Concept
- Reference

Within these topics, DITA specifies the information elements used to define the content, such as the title, paragraph, table, and list elements. For more information about DITA and using DITA source documents with ePublisher, see “Designing DITA Usage Standards”.

WebWorks Markdown

WebWorks Markdown as an input format is now supported by ePublisher and provides another way to include content within your published output.

Based on the community standard often referred to as GitHub Markdown, WebWorks Markdown provides many of the same capabilities as some of the other input formats, but without all their complexity.

Understanding Output Formats

The following sections summarize all default formats available in ePublisher and describes when it is best to use a particular format based on your desired output functionality. For more information about the requirements to view and develop each format, see “Requirements”.

ePublisher supports many output formats and you can implement many powerful features in your online content. Some features are available only in certain output formats. For more information, see “Features Available in Each Output Format” and “Supported Output Formats”.

Dynamic HTML

There are several HTML-related standards, such as HTML 3.2, HTML 4, and XHTML 1.0. HTML 4 is also referred to as Dynamic HTML (DHTML). In addition, there are multiple browsers and device types that can display content based on these standards. With all these variables, getting the content you need formatted the way you need it for your specific environment and requirements is critical. ePublisher provides the Simple HTML and the Dynamic HTML output formats to allow you to generate the HTML-based output you need. You can also customize this formatting to meet your requirements.

You can use the Dynamic HTML format to produce XHTML output that conforms to XHTML 1.0 standards and uses cascading style sheets that conform to the CSS1 standard. XHTML became a W3C recommendation in 2000. Dynamic HTML is recommended to produce output that you will publish on a Web server and provide to users running a Web browser, such as Firefox or Internet Explorer. You can also customize the Dynamic HTML output format to create a powerful, full-featured web site.

As the standards evolve, browsers and device platforms adjust to support the newer standards. With mobile devices, additional platforms and browsers have been introduced, which can complicate the decision about which standards your content can use.

The Simple HTML and Dynamic HTML output formats allow you to generate HTML content to integrate into your Web site. You can also create content for HTML-based release notes and content for hand-held devices, such as PDAs. These output formats provide the flexibility and control you need, with the ability to add a basic table of contents, index, and browse navigation. You decide whether to use all the aspects of DHTML and XHTML in the Dynamic HTML output format, or to use simplified HTML to support a wide range of browsers and platforms, including mobile devices and PDAs.

The Dynamic HTML output format produces HTML content that conforms to the HTML 4 and XHTML 1.0 standards, and uses cascading style sheets that conform to the CSS1 standard. Dynamic HTML (DHTML) is a collection of technologies developed to make HTML more dynamic and interactive. DHTML uses the following technologies to give the content developer control over the appearance and behavior of HTML elements in a browser window:

Static markup language (HTML 4)

HTML 4 extends HTML with mechanisms for style sheets, scripting, frames, embedding objects, improved support for right-to-left and mixed-direction text, richer tables, enhancements to forms, and improved accessibility for people with disabilities.

Presentation definition language (cascading style sheets)

Cascading style sheets (CSS) provide style definitions, such as fonts, colors, spacing, and positioning to HTML documents.

Client-side scripting language, such as JavaScript

JavaScript and other scripting languages provide compact, object-based scripting support for developing client and server Web applications.

Document Object Model

The Document Object Model provides a standard that allows programs and scripts to dynamically access, process, and modify the content of a page.

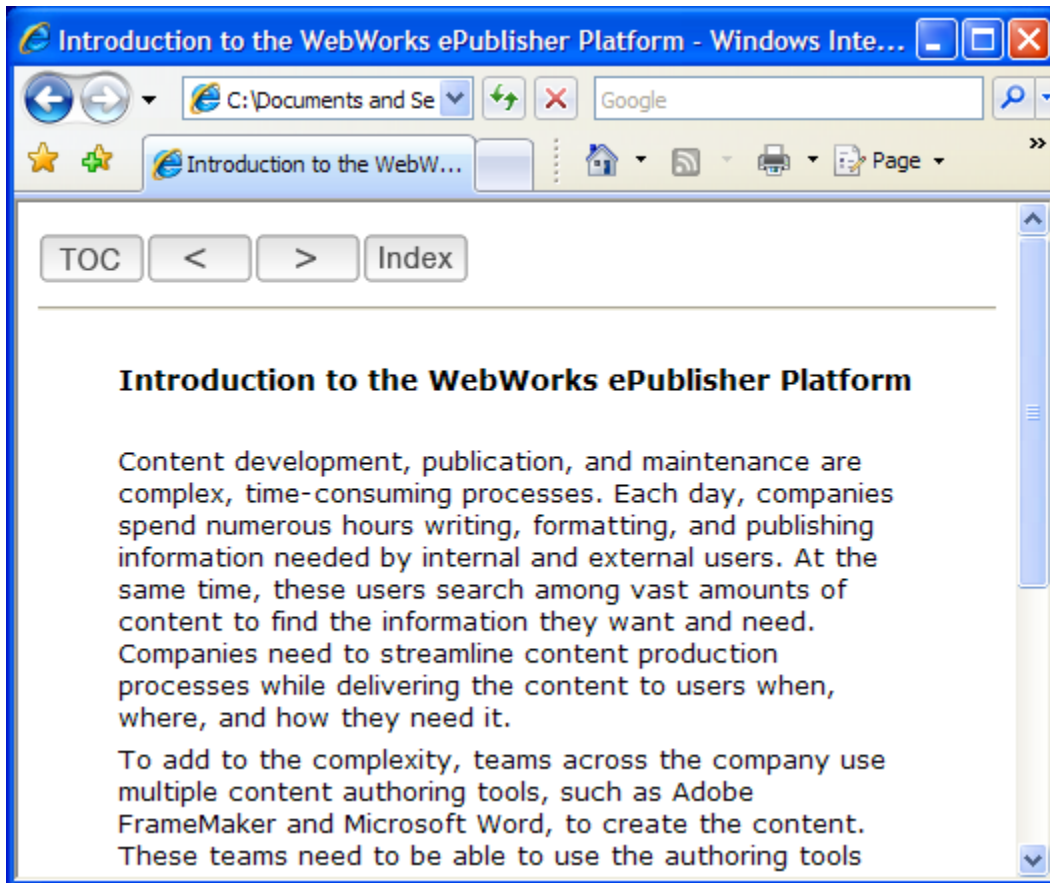
XHTML is an abbreviation for Extensible Hypertext Markup Language. XHTML 1.0 is similar to HTML 4, with tagging rules that conform to the requirements of XML. If you modify the page templates or styles in a Dynamic HTML project, make sure your changes conform to the XHTML requirements for future maintenance. However, as long as you create valid HTML, most current browsers can correctly display your output.

To determine whether the Dynamic HTML output format is what you need, review the following considerations:

- If all your users have current browsers and their viewing environment is not restricted, use the Dynamic HTML output format.
- If all your users have current browsers and you want to provide enhanced navigation controls, such as an expand/collapse table of contents, full text search, and related topics buttons, consider using the WebWorks Help or WebWorks Reverb output format.
- If you need to produce HTML content that can be viewed in older or less powerful browsers, or if you need to avoid using cascading style sheets (CSS), Java applets, and JavaScript, use the Simple HTML output format.

Dynamic HTML Output Viewer

Dynamic HTML does not provide a multi-pane viewer. This output format is displayed in a browser. By default, ePublisher adds a navigation bar at the top of the files it creates. ePublisher also creates a table of contents page and an index page.



You can specify whether to include the navigation bar at the top or the bottom of the page. You can also add company information to the page, and define the table of contents and index pages. For more information, see “Defining the Appearance of Pages”.

Delivering Dynamic HTML

When you generate output for your project, ePublisher creates the `Output` folder in your project folder, and then creates a folder in the `Output` folder for each generated target. Then, ePublisher creates a folder named for the project itself in the target folder. For example, when you generate output, ePublisher creates the following folder structure:

```
component Projects\projectname\Output\targetname\projectname
```

In this folder structure, *component* is the name of the ePublisher component you are using, such as ePublisher Express, *projectname* is the name of your ePublisher project, and *targetname* is the name of your ePublisher target, such as Dynamic HTML. To deliver your Dynamic HTML generated output, you need to deliver all the files and subfolders in the *targetname\projectname* folder.

The *targetname\projectname* folder contains the entry-point file, `toc.html` by default, which displays the table of contents. When the user opens the entry-point file, the browser uses all the files in the *targetname\projectname* folder to display the help, including the topic files and images.

eBook - ePub 2.0

ePublisher can deliver content to your mobile users and eReader platforms with the IDPF ePub 2.0 eBook standard. This format makes sense for long form content or reference materials which are not suitable for web delivery. eBooks enable users to access content offline, track their reading progress, and otherwise enjoy the benefits of traditional press books.

Keep in mind that eBooks primarily focus on readability. Therefore, eBooks lack support for content-sensitive help, JavaScript, and complex layouts. Tables are supported by the standard, yet you should carefully review the use of tables given the rendering limitations of current eReader platforms.

Consider using this format when delivering content to eReader devices, such as the Apple iPad.

Eclipse Help

To deliver help on multiple platforms, you need a solution that runs on all those platforms. This solution helps you avoid complications of each platform and allows you to deliver one solution that meets the needs on all your platforms. This common solution can save development time and effort.

ePublisher supports several Java-based help output formats, such as Eclipse Help, Oracle Help, and Sun JavaHelp, that provide a common solution for multiple platforms without JavaScript support. WebWorks Help and WebWorks Reverb also provide a cross-platform solution, but it requires JavaScript support. If you are delivering a Java-based application, or if your environment does not support JavaScript, a Java-based help solution can help you deliver your help content.

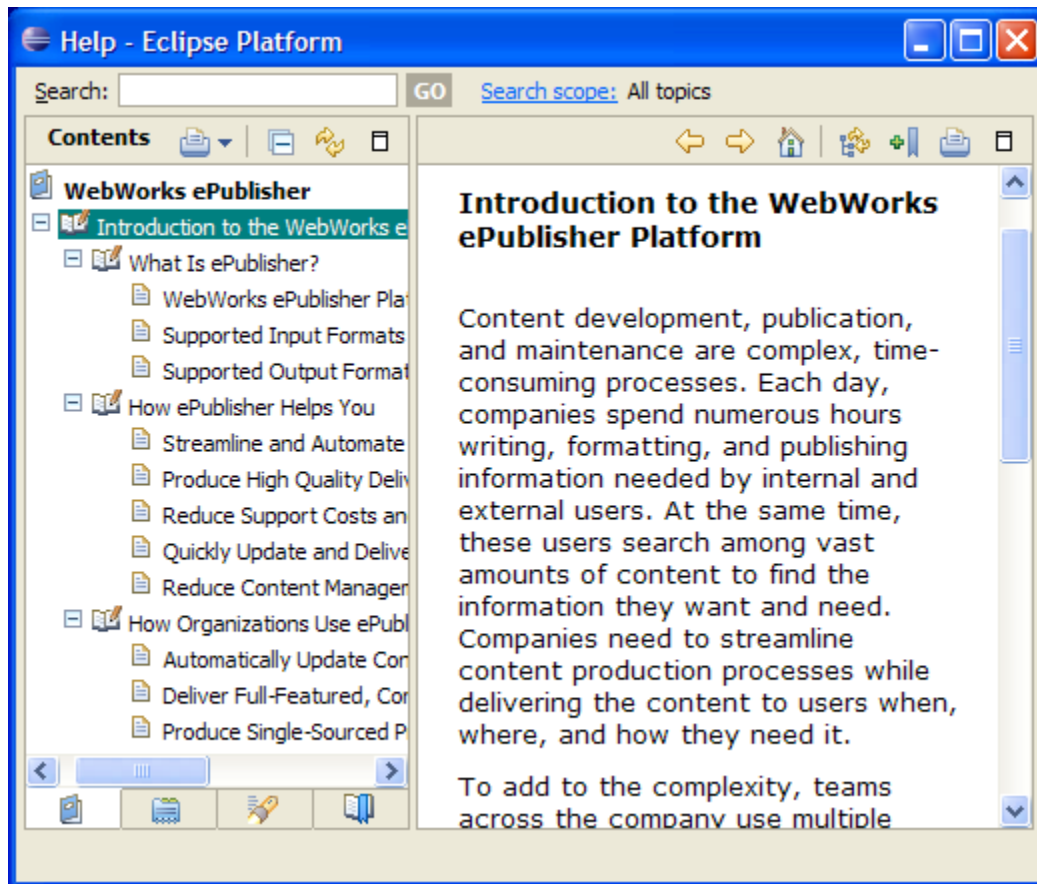
The Eclipse Help format uses a Java-based delivery environment to provide a comprehensive help viewer. Eclipse Help delivers content in HTML files and uses an XML-based table of contents. Once you install the Eclipse platform, you can view Eclipse Help files. ePublisher provides a standalone viewer for Eclipse Help so you can develop and view your Eclipse Help.

Eclipse Help requires the Eclipse integrated development environment (IDE), which makes it a good output format choice for products that already install and use the IDE. For more information about Eclipse Help, see the Eclipse SDK documentation at: help.eclipse.org/help33/index.jsp?topic=/org.eclipse.platform.doc.isv/guide/ua_help.htm.

You can also use Eclipse Help to deliver an Infocenter. You can use all the supported input formats to develop content for an Infocenter. Then, ePublisher allows you to publish that content in the required format. For more information about creating an Infocenter, see dita.xml.org/wiki/setting-up-the-eclipse-help-infocenter-for-publishing-dita-content.

Eclipse Help Viewer

The Eclipse Help viewer uses an embedded Apache Tomcat server. Similar to other help viewers, the Eclipse Help viewer provides a navigation pane with multiple tabs and a topic pane.



Delivering Eclipse Help

You can provide a help system in Eclipse Help format either as a collection of individual files or in a single, compressed Java archive `.jar` file. A `.jar` file, similar to a `.zip` file, compresses and stores a collection of files.

When you generate output for your project, ePublisher creates the `Output` folder in your project folder, and then creates a folder in the `Output` folder for each generated target. Then, ePublisher creates a folder named for the project itself in the target folder. For example, when you generate output, ePublisher creates the following folder structure:

```
component Projects\projectname\Output\targetname\projectname
```

In this folder structure, *component* is the name of the ePublisher component you are using, such as ePublisher Express, *projectname* is the name of your ePublisher project, and *targetname* is the name of your ePublisher target, such as Eclipse Help. To deliver your Eclipse Help generated output, you can provide the complete contents of the *targetname\projectname* folder to your Java application developers, and they can determine whether to deliver the `.jar` file or the individual files to application users.

Microsoft HTML Help

Delivering help information in a consistent manner on Windows computers was an important concern for Microsoft. Initially, Microsoft provided the WinHelp format for help content delivery. As the Windows platform advanced, Microsoft introduced the Microsoft HTML Help format. To help authors deliver content through the HTML Help Viewer provided with the Windows platform, Microsoft provides a help compiler and toolkit, HTML Help Workshop, that allows you to build Microsoft HTML Help `.chm` files.

The Microsoft HTML Help output format provides a standard help format for products on computers running the Windows operating system. The Microsoft HTML Help format delivers a single, compiled file that includes multiple source files compressed into one `.chm` file. Unlike the RTF-based Microsoft WinHelp format, the Microsoft HTML Help format is based on HTML/XHTML. Microsoft HTML Help is recommended to produce online help for 32-bit applications that run on a computer running Microsoft Windows 95 or later, including Windows Vista:

Microsoft HTML Help provides a comprehensive help format, including a table of contents, index, full-text search, and favorites in one integrated viewer window. The HTML Help Viewer uses standard Internet Explorer components and supports many Web technologies, such as HTML, ActiveX, Java, JavaScript, JScript and other scripting languages, and the Web image formats, such as `.gif`, `.jpg`, and `.png`.

In HTML Help, writers list a collection of source files in a help project `.hhp` file with other project-related settings. The writers then use HTML Help Workshop to compile the help and create the `.chm` file. During compilation, HTML Help Workshop uses the help project `.hhp` file to determine how HTML topic files, image files, contents `.hhc` files, index `.hhk` files, and any other elements appear in the single, compressed help file. ePublisher automates this process and integrates it into help generation.

Benefits of Microsoft HTML Help

The Microsoft HTML Help output format generates output files that conform to the HTML 4 and XHTML 1.0 standards. The content uses Cascading Style Sheets that conform to the CSS1 standard. This output format produces all the files required to create the Microsoft HTML Help .chm file.

ePublisher streamlines the help generation process. ePublisher uses your Stationery and project settings to transform your source documents to the input files needed by Microsoft HTML Help Workshop. Then, ePublisher uses the help compiler to build the help project and create the .chm file. This integrated process simplifies the authoring environment, but it also creates all the underlying files to help you customize the process, if needed.

Microsoft HTML Help provides several important benefits:

- You have a single, compressed .chm output file to deliver.
- Your audience can read your content in the standard Windows HTML Help Viewer.
- The HTML Help Viewer provides a powerful search engine that supports partial words, synonym searching, and other advanced search options.

Restrictions and Requirements for Microsoft HTML Help

The Microsoft HTML Help output format is recommended to produce standard help for 32-bit applications that run on a computer running Windows 95 or later. For more information, see “Requirements”. Review the following considerations when deciding whether to deliver HTML Help:

- Your audience cannot read the .chm file over a network or over the Web. The .chm file must be installed on the local computer.
- Some security settings can interfere with .chm file use.
- A .chm file supports many Web technologies, such as scripting languages, which also allow them to contain and transport viruses and security risks. For this reason, many email systems remove .chm files when attached to an email message.

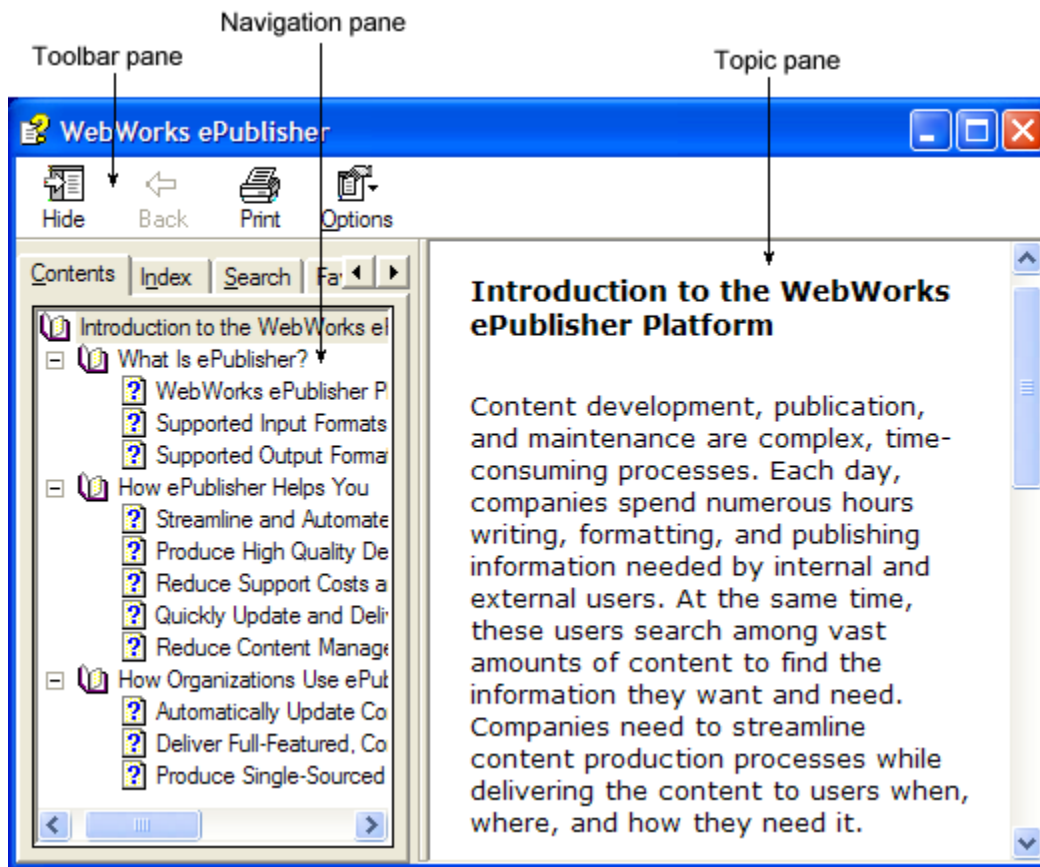
HTML Help Viewer

When you open a Microsoft HTML Help .chm file, Windows displays the content in the Help Viewer window. Microsoft HTML Help Workshop allows the writer to compile and customize the .chm file. ePublisher uses your source documents to generate all the files needed to compile the .chm file.

By default, the tripane HTML Help Viewer includes a table of contents, index, full-text search, and favorites in one, integrated viewer window. You can customize this viewer window. The user also has options to show or hide the navigation pane, based on the options you defined.

The default tripane view includes the following panes:

- The **toolbar pane** displays buttons that provide additional functions and navigation options to the user. HTML Help Workshop allows you to select which buttons to include in your help file.
- The **navigation pane** displays tabs to access the table of contents, index, full-text search, and favorite topics. HTML Help Workshop allows you to select which tabs to include in your help file.
- The **topic pane** displays the information contained in the source documents.



Toolbar Pane in HTML Help

The toolbar pane provides several buttons that provide additional functions, such as returning to the previously viewed topic, showing and hiding the navigation pane, and printing a help topic. The writer can customize which buttons are included in the toolbar pane through HTML Help Workshop. The following table lists the supported buttons.

Button	Description
Hide/Show	Opens or closes the Navigation pane.
Back	Displays the previously viewed topic like the Back button in a browser.
Forward	Displays the previously viewed topic if the user clicked the Back button. This button functions like the Forward button in a browser.
Stop	Stops retrieving the file information and contents.
Refresh	Updates the topic that is currently displayed in the topic pane.
Home	Displays the default topic you defined for the help file.
Options	Opens a menu that provides many commands, such as Home, Show, Back, Stop, Refresh, Print, Search Highlight On/Off, and Internet Options. This menu also includes commands for all the buttons included on help windows.
Print	From the Contents tab, this button allows the user to print the selected topic, and optionally all subtopics. From the Index or Search tab, opens the Print dialog box to print the current topic.
Locate	Displays in the table of contents the location of the current topic is not listed in the table of contents, this command will not work.
Jump 1	Jumps to an author-designated Web page or help topic.
Jump 2	Jumps to an author-designated Web page or help topic.

Navigation Pane in HTML Help

By default, the navigation pane provides the following tabs, which you can include or exclude from your help output through HTML Help Workshop:

Contents tab

Displays the table of contents in the form of a expand/collapse tree view. The table of contents includes all paragraph styles that you assigned a TOC level in Style Designer. When the user selects an entry in the Contents tab, the information from that topic is displayed in the topic pane. You can customize the icons used for specific topics within the table of contents.

Index tab

Displays an alphabetical list of keywords associated with topics in the source documents. Writers use their source document authoring tool to create standard index markers or field codes that define these keywords in their source documents.

Search tab

Provides a powerful full-text search feature with several advanced search options. The user can type a search string and then select **Display** to view a list of topics that contain the word or phrase specified.

Favorites tab

Displays a list of topics in the help that the user has added to his or her personal list of favorites. This tab allows users to bookmark help topics they often use or want to quickly find in the future. When the user selects a topic on this tab, the information from that topic is displayed in the topic pane.

Topic Pane in HTML Help

The output pages generated from your source documents are displayed in the topic pane. When a user selects a topic on any of the tabs in the navigation page, the content of that topic is displayed in the topic pane.

Topic Only View in HTML Help

The show/hide button in the toolbar pane allows users to add or remove the navigation pane. The writer can also define a window in HTML Help Workshop that does not include the navigation pane, or with the navigation pane hidden by default. Without the navigation pane, more screen area is available to display the help topic content, or for the application itself. However, the navigation pane helps users view where they are in the organization of topics, and quickly browse and access other topics in the table of contents.

Understanding HTML Help Workshop

HTML Help Workshop is a help authoring tool that allows you to create and manage Microsoft HTML Help projects and their related files. You can use this tool, which is installed with ePublisher, to further customize your help, such as changing the appearance of the Contents and Search tabs. You can also create an override for your project .hhp file in your ePublisher project, which allows you to implement the custom HTML Help Viewer window you need. For more information about Microsoft HTML Help and its related files and customization options, see the help for HTML Help Workshop.

HTML Help Project File (.hhp)

The HTML Help project `.hhp` file identifies all the elements of an HTML Help project. This project file is separate from the ePublisher project. The HTML Help project file contains the information HTML Help Workshop needs to combine the source files, images, index, and table of contents into a single, compiled help `.chm` file.

The HTML Help project file also defines the appearance and behavior of the HTML Help Viewer window. ePublisher creates the HTML Help project file based on the `template.hhp` file and the settings in your ePublisher project. You can override the default `template.hhp` file to adjust the default appearance of your HTML Help, such as the default size and position of the HTML Help Viewer window or the buttons displayed in the toolbar pane. You can also define additional windows in the `template.hhp` file, such as a window without the navigation pane. For more information, see “Adjusting the HTML Help Viewer Window Size and Toolbar Buttons” and “Creating an Additional HTML Help Window Definition”.

HTML Help Contents File (.hhc)

The HTML Help contents `.hhc` file defines the table of contents for the help. This file includes entries for each occurrence of the paragraph styles for which you assigned a TOC level in your ePublisher project. HTML Help displays the contents of this file on the Contents tab of the navigation pane. ePublisher generates the `toc.hhc` file for your project using the `template.hhc` file.

HTML Help Index File (.hhk)

The HTML Help index `.hhk` file defines the index for the help. This file includes entries based on the index markers and field codes in your source documents. HTML Help displays the contents of this file on the Index tab of the navigation pane. ePublisher generates the `index.hhk` file for your project using the `template.hhk` file.

HTML Help Mapping File (.h)

The HTML Help mapping `.h` file identifies the mapping information for context-sensitive help links. This file includes an entry for each TopicAlias marker or field code in your source documents. Application developers build this file with their project and use the defined aliases to display the appropriate topic in the help. This file is linked in the `MAP` section of the `.hhp` file, and its entries coordinate with the entries in the `ALIAS` section of the `.hhp` file.

Delivering HTML Help

When you generate output for your project, ePublisher creates the `Output` folder in your project folder, and then creates a folder in the `Output` folder for each generated target. Then, ePublisher creates a folder named for the project itself in the target folder. For example, when you generate output, ePublisher creates the following folder structure:

```
component Projects\projectname\Output\targetname\projectname
```

In this folder structure, *component* is the name of the ePublisher component you are using, such as ePublisher Express, *projectname* is the name of your ePublisher project, and *targetname* is the name of your ePublisher target, such as Microsoft HTML Help 1.x.

By default, ePublisher compiles the `.chm` and stores it in the *targetname* folder. To deliver your HTML Help, you need to deliver the `.chm` file.

If you created context-sensitive help, your application developer needs to build with the *targetname**projectname**projectname*.h file. If you created What's This field-level help, your application developer also needs to build with the *targetname**projectname*\whatisthis.h file. For more information about these mapping files, see “Using Context-Sensitive Help in HTML Help” and “Defining What's This (Field-Level) Help in HTML Help”.

Oracle Help

To deliver help on multiple platforms, you need a solution that runs on all those platforms. This solution helps you avoid complications of each platform and allows you to deliver one solution that meets the needs on all your platforms. This common solution can save development time and effort.

ePublisher supports several Java-based help output formats, such as Eclipse Help, Oracle Help, and Sun JavaHelp, that provide a common solution for multiple platforms without JavaScript support. WebWorks Help and WebWorks Reverb also provide a cross-platform solution, but it requires JavaScript support. If you are delivering a Java-based application, or if your environment does not support JavaScript, a Java-based help solution can help you deliver your help content.

The Oracle Help format is a Java-based help format that produces the complete set of files required to deliver online help based on the Oracle Help for Java technology. Oracle Help for Java is a set of Java components and an API for developing and displaying HTML-based help content in a Java environment. Oracle Help is recommended to produce help for an application written in the Java programming language.

Note: Oracle also offers a separate technology called Oracle Help for the Web. ePublisher does not support Oracle Help for the Web.

To view Oracle Help, users must have a Java Virtual Machine (JVM) installed. If you produce help for a Java application, the JVM will be installed on computers running the application. The Oracle Help components must also be installed on the computer.

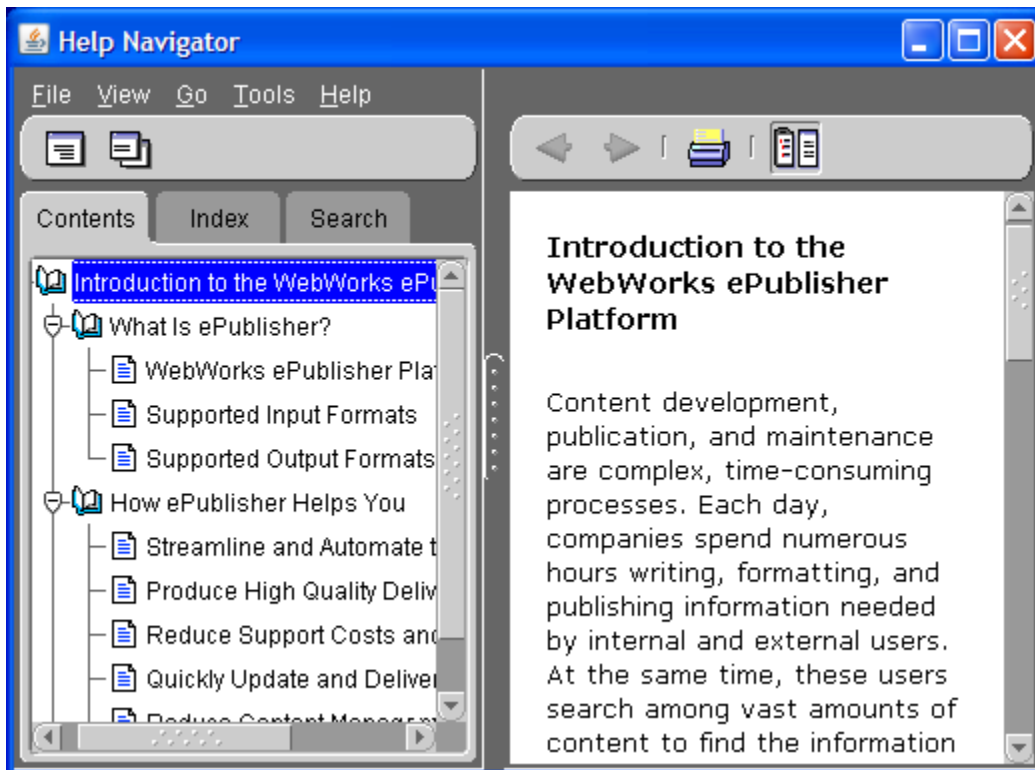
The Oracle Help output format produces content that conforms to the HTML 4 and XHTML 1.0 standards and uses Cascading Style Sheets that conform to the CSS1 standard. This output format also produces files that are specifically required for Oracle Help. For more information about the Oracle Help technology, see the Oracle Web site at www.oracle.com/technology/tech/java/help.

When deciding whether to use the Oracle Help output format, review the following considerations:

- Oracle Help is one of the output formats you can use to produce help for a Java application. You can also use the Oracle Help output format to produce content for the Web if all your users have, or are prepared to install, a Java Virtual Machine.
- If you need to produce help for a pure Java application, you may also consider using the Eclipse Help and Sun JavaHelp output formats. Each output format has slightly different requirements, appearance, and behavior.
- If you need to produce help for a Web-based application hosted on a Web server or on an intranet server that is not a pure Java application, you may consider using the WebWorks Help or WebWorks Reverb output format.

Oracle Help Viewer

The Oracle Help output format produces Java-based online help. Oracle Help includes a navigation pane and a topic pane that is similar to other help viewers.



Understanding Oracle Help Files

Oracle Help has several files that work together to define and create the Oracle Help solution. ePublisher incorporates these files into the compressed helpset file, so you do not need to distribute them to users.

Helpset .hs File in Oracle Help

The helpset `.hs` file contains descriptive information about your Oracle Help helpset. ePublisher generates this file using the `template.hs` page template. For more information about the helpset file, see the Oracle Help documentation.

Control .xml Files

The control .xml files define the table of contents, index, and topic aliases in your Oracle Help project. These files are named *projectTOC.xml*, *projectMap.xml*, *projectLinks.xml*, and *projectIndex.xml*, where *project* is the name of your ePublisher project. For more information about these files, see the Oracle Help documentation.

Full Text Search .idx Index File

The internal full text search `.idx` index file provides support for the full text search feature.

Manifest .mft File

ePublisher automatically generates the manifest `.mft` file. This internal file is used to create the final Java archive `.jar` file that you distribute to users.

Delivering Oracle Help

You can provide a help system in Oracle Help format either as a collection of individual files or in a single, compressed Java archive `.jar` file. A `.jar` file, similar to a `.zip` file, compresses and stores a collection of files.

When you generate output for your project, ePublisher creates the `Output` folder in your project folder, and then creates a folder in the `Output` folder for each generated target. Then, ePublisher creates a folder named for the project itself in the target folder. For example, when you generate output, ePublisher creates the following folder structure:

```
component Projects\projectname\Output\targetname\projectname
```

In this folder structure, *component* is the name of the ePublisher component you are using, such as ePublisher Express, *projectname* is the name of your ePublisher project, and *targetname* is the name of your ePublisher target, such as Oracle Help. To deliver your Oracle Help generated output, you can provide the complete contents of the *targetname\projectname* folder to your Java application developers, and they can determine whether to deliver the `.jar` file or the individual files to application users.

PDF

The PDF format creates a PDF file for each of your source files, a single PDF file containing all source files within a group, or both. PDF is recommended to produce a print-ready output file. This format allows you to create and deliver a single file that your users can review online, and print if and when needed.

PDF - XSL-FO

The PDF - XSL-FO format creates a PDF file for each of your source files, a single PDF file containing all source files within a group, or both. PDF - XSL-FO is recommended for producing a PDF output file that conforms to the settings you configure in ePublisher. These settings can be controlled independent of your source file's authoring environment or they can be used to work in conjunction with your source file's paragraph and character level settings. This format allows you to create and deliver PDF files that conform to the style and layout settings that you can centrally control in ePublisher. This format may be especially useful when mixing documents from different input formats or different layouts and/or styles.

Sun JavaHelp

To deliver help on multiple platforms, you need a solution that runs on all those platforms. This solution helps you avoid complications of each platform and allows you to deliver one solution that meets the needs on all your platforms. This common solution can save development time and effort.

ePublisher supports several Java-based help output formats, such as Eclipse Help, Oracle Help, and Sun JavaHelp, that provide a common solution for multiple platforms without JavaScript support. WebWorks Help and WebWorks Reverb also provide a cross-platform solution, but it requires JavaScript support. If you are delivering a Java-based application, or if your environment does not support JavaScript, a Java-based help solution can help you deliver your help content.

The Sun JavaHelp format produces help systems for applications written in the Java programming language. Sun JavaHelp uses HTML files and cascading style sheets, but Sun JavaHelp is not usually displayed in a Web browser. Sun JavaHelp users usually view the help system in the Sun JavaHelp viewer application or in a custom viewer application created by your application developers.

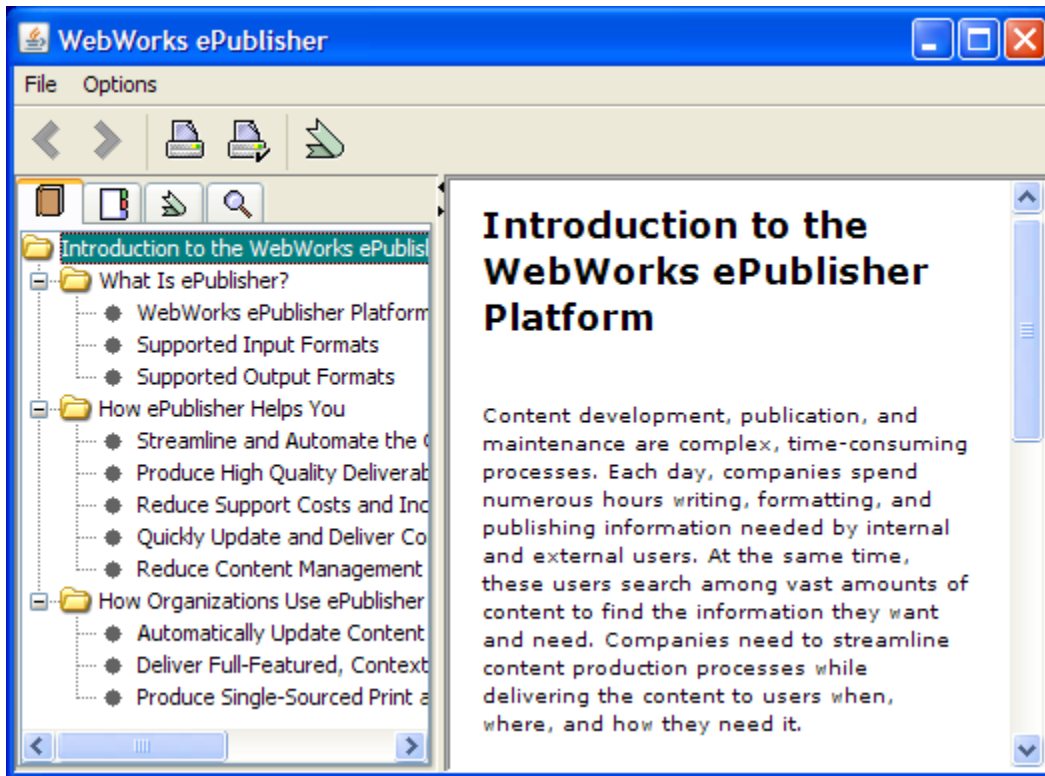
Sun JavaHelp is recommended to create help for an application written in the Java programming language. Sun JavaHelp includes a navigation pane with Contents, Index, and Search tabs, and a topic pane that displays the actual help content topics.

The Sun JavaHelp output format delivers online help for a Java application and has no dependency on JavaScript. When deciding whether to use the Sun JavaHelp output format, review the following considerations:

- Sun JavaHelp is one of the output formats you can use to produce help for a Java application.
- If you need to produce help for a pure Java application, you may also consider using the Eclipse Help and Oracle Help output formats. Each output format has slightly different requirements, appearance, and behavior.
- If you are creating help for a Java application and you can also use JavaScript, you may consider using the WebWorks Help or WebWorks Reverb output formats.
- If you need to produce help for a Web-based application, hosted on a Web server or on an intranet server, that is not a pure Java application, you may consider using the WebWorks Help or WebWorks Reverb output formats.

Sun JavaHelp Viewer

Similar to other help systems, Sun JavaHelp provides a navigation pane and a topic pane. The navigation pane provides Contents, Index, Favorites, and Search tabs. The topic pane displays the content of the help topic selected on a tab in the navigation pane.



Understanding Sun JavaHelp Files

Sun JavaHelp delivers content in HTML files. The table of contents, index, and helpset .hs files use the Extensible Markup Language (XML) format.

Helpset .hs File in Sun JavaHelp

The helpset .hs file contains configuration information in XML format that Sun JavaHelp needs to display your help system. The following figure shows a sample helpset file:

```
<?xml version='1.0' encoding='ISO-8859-1' ?>

<!DOCTYPE helpset PUBLIC "-//Sun Microsystems Inc.//DTD JavaHelp HelpSet Version 1.0//EN"
"http://java.sun.com/products/javahelp/helpset_1_0.dtd">

<helpset version="1.0">

  <title>Interesting help project</title>

  <maps>

    <homeID>home</homeID>

    <mapref location="test.jhm" />

  </maps>

  <view>

    <name>TOC</name>

    <label>Interesting help project</label>

    <type>javax.help.TOCView</type>

    <data>testt.xml</data>

  </view>

  <view>

    <name>Index</name>

    <label>Index</label>

    <type>javax.help.IndexView</type>

    <data>testi.xml</data>

  </view>

  <view>

    <name>Search</name>

    <label>Search</label>

    <type>javax.help.SearchView</type>

    <data engine="com.sun.java.help.search.DefaultSearchEngine">

      JavaHelpSearch

    </data>

  </view>

</helpset>
```

```
</data>
```

```
</view>
```

```
</helpset>
```

The helpset file begins with an XML declaration. The next part of the file, as shown in the following sample, defines the help version, the title of the help, and the location of the mapping file that is used to support context-sensitive help:

```
<helpset version="1.0">
```

```
<title>Interesting help project</title>
```

```
<maps>
```

```
<homeID>home</homeID>
```

```
<mapref location="test.jhm" />
```

```
</maps>
```

The remaining sections of the file define several views, such as the table of contents (TOC), index, and search. Each `<view>` tag includes several tags that define the view, such as the `<name>` tag that specifies the name of the view and the `<data>` tag that specifies the data file for the view.

You can also define your own views in Sun JavaHelp by overriding the `template.hs` file. For example, you can create an alternate list of topics and provide that list as a separate view.

Contents toc.xml File in Sun JavaHelp

The contents `toc.xml` file defines the items in the table of contents as shown in the following example file:

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
```

```
<!DOCTYPE toc PUBLIC "-//Sun Microsystems Inc.//DTD JavaHelp TOC Version 1.0//EN" "http://  
java.sun.com/products/javahelp/toc_1_0.dtd">
```

```
<toc version="1.0">
```

```
<tocitem target="tutorial_htm_1003580" text="Hybrid Web/CD:">
```

```
<tocitem target="tutorial_htm_1004263" text="Roadmap"/>
```

```
<tocitem target="tutorial_htm_1005107" text="Explosion"/>
```

```
<tocitem target="tutorial_htm_999374" text="Bandwidth"/>
```

```
...
```

```
</tocitem>
```

```
</toc>
```

Index ix.xml File in Sun JavaHelp

The index `ix.xml` file lists your index entries as shown in the following example file:

```
<?xml version='1.0' encoding='ISO-8859-1' ?>

<!DOCTYPE index PUBLIC "-//Sun Microsystems Inc.//DTD JavaHelp Index Version 1.0//EN"
"http://java.sun.com/products/javahelp/index_1_0.dtd">

<index version="1.0">

  <indexitem target="tutor4_htm_999680" text="1996 Telecom Act"/>

  <indexitem target="tutor11_htm_999661" text="active catalog"/>

  <indexitem target="tutor10_htm_999632" text="ATM"/>

  <indexitem target="tutor10_htm_999634" text="audio synchronization"/>

  <indexitem text="bandwidth">

    ...

  </index>
```

Map .jhm File in Sun JavaHelp

Sun JavaHelp supports context-sensitive help through a mapping `.jhm` file. The mapping file contains information that links your Sun JavaHelp topics to particular locations in the Java application. For more information, see “Using Context-Sensitive Help in Oracle Help and Sun JavaHelp”.

Delivering Sun JavaHelp

You can provide a help system in Sun JavaHelp format either as a collection of individual files or in a single, compressed Java archive `.jar` file. A `.jar` file, similar to a `.zip` file, compresses and stores a collection of files.

When you generate output for your project, ePublisher creates the `Output` folder in your project folder, and then creates a folder in the `Output` folder for each generated target. Then, ePublisher creates a folder named for the project itself in the target folder. For example, when you generate output, ePublisher creates the following folder structure:

```
component Projects\projectname\Output\targetname\projectname
```

In this folder structure, *component* is the name of the ePublisher component you are using, such as ePublisher Express, *projectname* is the name of your ePublisher project, and *targetname* is the name of your ePublisher target, such as JavaHelp. To deliver your Sun JavaHelp generated output, you can provide the complete contents of the *targetname\projectname* folder to your Java application developers, and they can determine whether to deliver the `.jar` file or the individual files to application users.

WebWorks Help

Delivering help content on multiple platforms and browsers can be a challenge. You need a solution that provides advanced help features, such as full-text search, expand/collapse sections, browse and breadcrumb navigation, related topics, and a table of contents tree control. WebWorks Help delivers these features and others on multiple platforms and browsers. Whether your help needs to be viewed on the Macintosh or Windows operating system, displayed with Firefox, Safari, or Internet Explorer, WebWorks Help provides the flexible, adaptable help solution you need.

The WebWorks Help format combines standard XHTML, CSS1, and CSS2 with JavaScript to provide many online help navigation controls and features, including an expandable/collapsible table of contents, an index, full text search, and Related Topics menus. You can easily integrate WebWorks Help with applications written in C, C++, Java, or Visual Basic, as well as Web-based applications. This comprehensive help format allows you to quickly create and deploy context-sensitive help and comprehensive help that you can fully customize to match your corporate branding.

WebWorks Help enables you to publish your content in a variety of ways:

- On a Web server
- On a network server available to multiple users through a share
- On the local computer where the application is installed
- On a CD-ROM

The Frameset View in WebWorks Help

When you open WebWorks Help, you can choose whether to display the navigation pane and the content, or only the content in the browser. You can control the way WebWorks Help opens based on whether the user selected a context-sensitive help link or opened the comprehensive help. This flexibility allows you to deliver your content the way the user needs it.

The frameset view divides the browser window into several areas:

Navigation pane

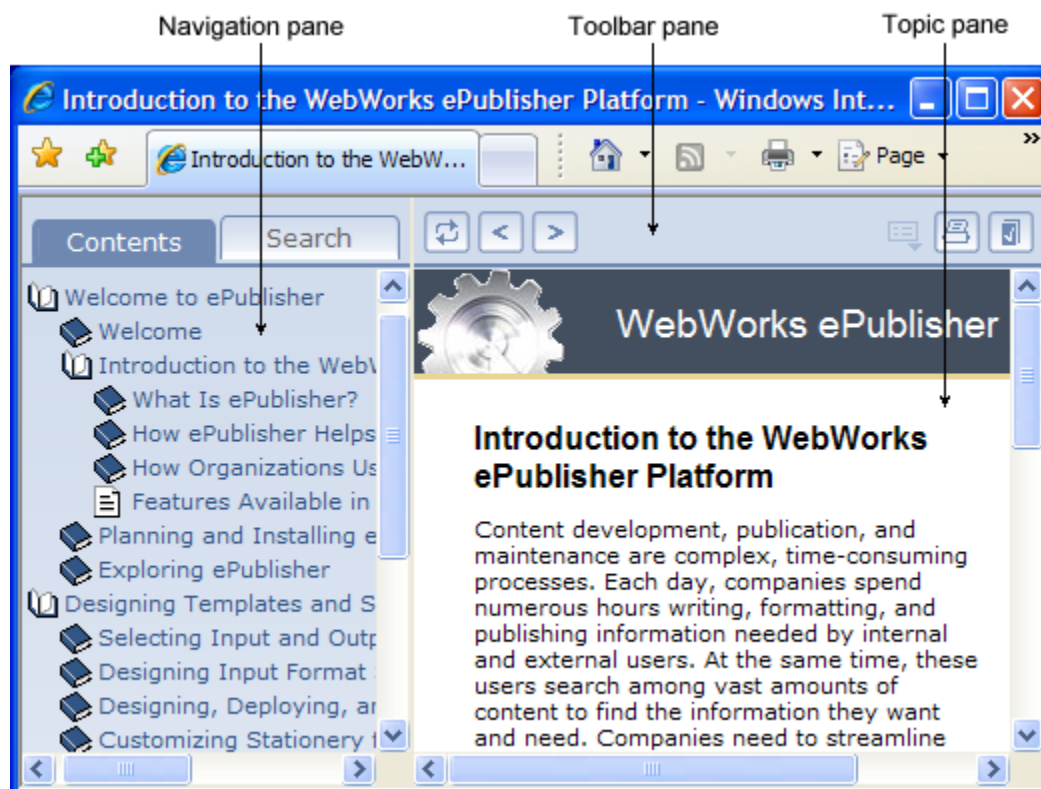
Provides multiple tabs for navigation elements, such as the table of contents and full-text search.

Toolbar pane

Provides several standard toolbar buttons to perform common tasks, such as show or hide the navigation pane, browse to the next topic, and print the content.

Topic pane

Displays the content of the topic selected in the navigation pane.



Navigation Pane in WebWorks Help

By default, the navigation pane displays several tabs that provide core navigational features for your help:

Contents tab

Displays the table of contents, including entries for each style that you assigned a TOC level value in Style Designer. When a user clicks on item in the Contents tab, the selected topic is displayed in the topic pane. This tab displays the table of contents as an expand/collapse tree view. Each book icon represents a table of content entry that has subentries.

Index tab

Displays an alphabetical list of keywords associated with topics. To view index entries, select a letter to display the entries that start with that letter. When the user clicks on an index entry, the related topic is displayed in the topic pane. The writer defines the keywords as index entries in the source documents.

Search tab

Provides a full-text search. When the user clicks **Go**, WebWorks Help lists the titles of the topics whose content contains the words specified in the **Search** field. This tab provides **Rank** and **Title** columns. Each listed topic has a relevancy ranking number, which reflects how well the topic matches the search criteria. The ranking is specified in the `wwh_files.xml` file. For more information about how to modify the ranking, see “Modifying the Search Ranking”.

Favorites tab

Lists the topics that the current user added to his or her list of personal favorites. In WebWorks Help, users can add frequently accessed or important help topics to their personal list of favorites. When the user clicks on a topic on the Favorites tab, the help topic is displayed in the topic pane. The Favorites tab also provides a Remove button that allows users to delete any unwanted topics from their list.

Toolbar Pane in WebWorks Help

By default, the toolbar pane displays a set of buttons that provide users with additional features. These buttons allow users to navigate through the help and access common functions, such as printing the displayed topic and emailing a link to a topic. The standard WebWorks Help toolbar includes the following buttons:

Show in Contents button

Allows users to locate the topic they are viewing in the table of contents. When a user clicks this button, WebWorks Help highlights the entry in the Contents tab that corresponds to the currently displayed topic.

Show Navigation button

Allows users to show the hidden navigation pane. When a context-sensitive help link displays a topic with the navigation pane hidden, this button is displayed in the toolbar. When a user clicks this button, WebWorks Help displays the navigation pane and highlights the entry in the Contents tab that corresponds to the currently displayed topic.

Previous button

Allows users to navigate back to topics that precede the currently displayed topic in the help.

Next button

Allows users to navigate forward to topics that follow the currently displayed topic in the help.

PDF button

Displays a PDF file of the source document from which the currently displayed topic was generated.

Related Topics button

Displays a list of topics that share a common or related theme with the currently displayed topic. The writer must define related topics in the source documents and the Stationery designer must enable related topics support for the help to display them. For more information, see “Defining Related Topics”.

Print button

Prints the currently displayed topic.

Email button

Allows you to collect feedback from your users. This button opens a blank email message addressed to the email address you specify. The subject line of the email message identifies the displayed topic when the user clicked the Email button.

Topic Pane in WebWorks Help

This pane displays the content of the help topics generated from your source documents. When a user selects a topic on the Contents, Search, Index, or Favorites tab, the topic pane displays the content of that topic.

Topic Only View in WebWorks Help

You can display a topic-only view, which hides the navigation pane and displays only the toolbar pane and the topic pane. With this topic-only view, an application can open a smaller browser window that leaves more area on the screen for the application itself. Context-sensitive help often uses the topic-only view.

WebWorks Help Output Files

When you generate the WebWorks Help output format, ePublisher creates all the files required to deliver and display WebWorks Help. The `index.html` file is the default entry file that defines the frameset used by the help. To open the help, users open this entry file. ePublisher creates the `.html`, `.css`, `.js`, and various image files included in the WebWorks Help output.

Delivering WebWorks Help

When you generate output for your project, ePublisher creates the `Output` folder in your project folder, and then creates a folder in the `Output` folder for each generated target. Then, ePublisher creates a folder named for the project itself in the target folder. For example, when you generate output, ePublisher creates the following folder structure:

```
component Projects\projectname\Output\targetname\projectname
```

In this folder structure, *component* is the name of the ePublisher component you are using, such as ePublisher Express, *projectname* is the name of your ePublisher project, and *targetname* is the name of your ePublisher target, such as WebWorks Help. To deliver your WebWorks Help generated output, you need to deliver all the files and subfolders in the *targetname\projectname* folder.

The *targetname\projectname* folder contains the entry-point file, `index.html` by default, which establishes the help set appearance. When the user opens the entry-point file, the browser uses all the files in the *targetname\projectname* folder to display the help, including all the topic files, generated `.css` files, `.pdf` files, images, and WebWorks Help components.

Searching WebWorks Help

Use the following guide to assist users in finding terms in a WebWorks Help 5.0 help system:

Boolean

All search words and phrases have an implicit AND

Word Search

For example, searching: `eggs bacon` returns all documents containing “eggs” and “bacon”

Phrase Search

For example, searching “`Good Morning`” returns all documents containing “good” and “morning” where the two words are adjacent

Wildcard

For example, searching `e*` returns all documents containing words that start with e.

To further customize the way the search results are displayed, See [Modifying the Search Ranking](#)

WebWorks Reverb

WebWorks Reverb has many of the features found in WebWorks Help, and goes one step further by providing high-performance load times combined with optimal handling of the end-user's device resolution. This means that users viewing this type of output on a mobile device will be able to view the files in a fast, light-weight manner that is optimal for lower resolutions and/or touch operated screens. Furthermore, users on a more traditional higher-resolution computer monitor will still have the same advantages offered by a complete online help system. In other words, Reverb adapts to the traits of the device that is doing the reading of the content.

WebWorks Reverb also offers a commenting and end-user feedback mechanism using the Disqus commenting and discussion platform. This platform is a leading solution on the Internet for comment handling and has no cost and provides automatic hosting of your end-users comments in a way that is transparent and effective.

If you are deploying your Reverb output on a Microsoft IIS web server then you can use the Microsoft IIS Search alternative, otherwise you can use the Client-side Search alternative which will work on any web server.

Reverb also provides easy Google Analytics integration. Using Google Analytics in your web files will be very easy to configure with this format.

If you have end-users that require localized help files, WebWorks Reverb offers a seamless integration with the Google Translate Element. This feature is very powerful and can be used instead of an expensive and burdensome set of translated files and help volumes, which requires a separate set of content for each language that you are supporting. Using WebWorks Reverb, you can now create instantaneously localized help volumes without any translation requirements and only one set of help files.

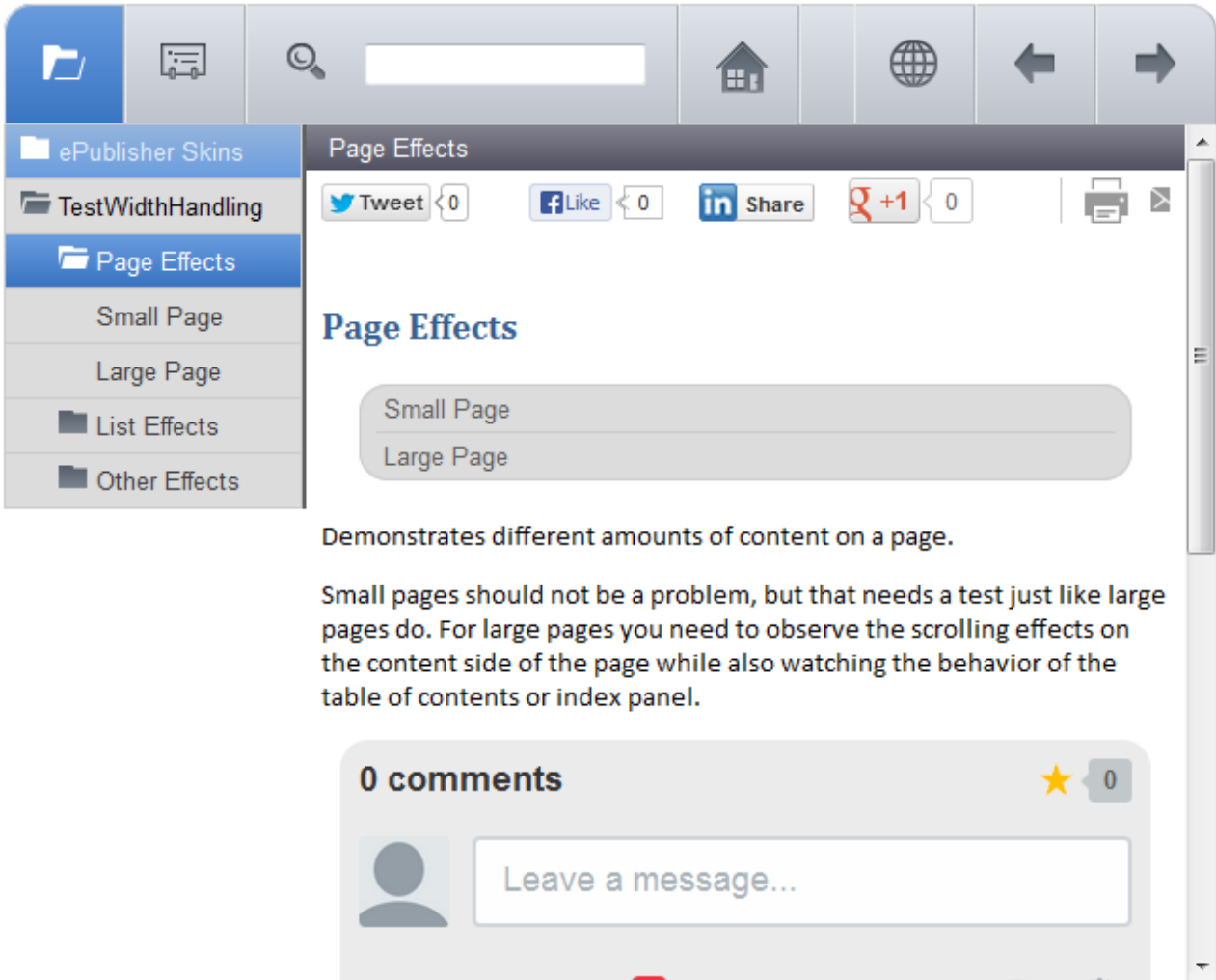
Choosing a Skin in WebWorks Reverb

You can select from a number of predefined skins that are available for the WebWorks Reverb output format. Each skin has been professionally designed with assistance from both graphic designers and web developers so that your users can get an ideal experience when browsing your documentation.

You may select among the following types of skins:

Classic

The original WebWorks Reverb skin. Same look and feel with the latest features of Reverb. Using a neutral gray color scheme along with a combination of gradients, multi-colored icons, and rounded corners this skin is great for all around acceptability and use across any platform and website.



Corporate

Modeled after high profile technology websites, the *Corporate* skin will give your online help a polished look. Another feature of the *Corporate* skin is that it provides an additional area to display your company name or organizational branding above the table of contents, index, and search panels.

Corporate Skin: Color Schemes
Blue

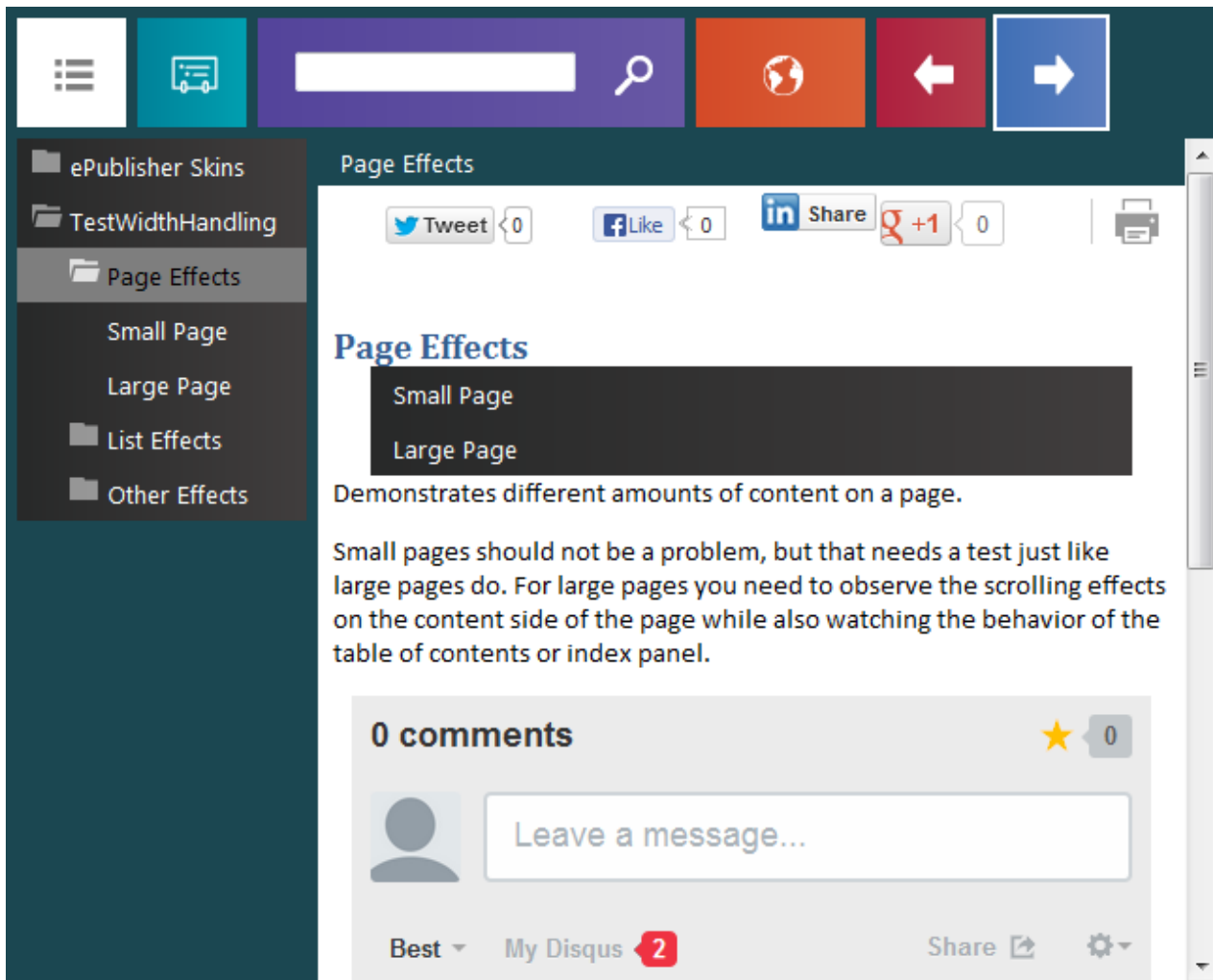
Corporate Skin: Color Schemes
Grey
Red

The screenshot shows a web application interface. At the top is a red navigation bar with icons for a menu, a printer, a globe, and navigation arrows, along with a search bar. On the left is a sidebar menu with a red header 'Company Name' and items: 'ePublisher Skins', 'TestWidthHandling', 'Page Effects' (selected), 'List Effects', and 'Other Effects'. The 'Page Effects' sub-menu is open, showing 'Small Page' and 'Large Page'. The main content area has a red header 'Page Effects' and a list of 'Small Page' and 'Large Page'. Below this is a text block: 'Demonstrates different amounts of content on a page. Small pages should not be a problem, but that needs a test just like large pages do. For large pages you need to observe the scrolling effects on the content side of the page while also watching the behavior of the table of contents or index panel.' At the bottom is a comments section with '0 comments', a star icon, a '0' in a speech bubble, a user profile icon, and a 'Leave a message...' text input field.

Metro

The *Metro* skin represents the latest ideas in computer interface design and comes straight from the *Metro Design Language* now the heart of interfaces used in all Microsoft desktop, smartphone, and game box operating platforms.

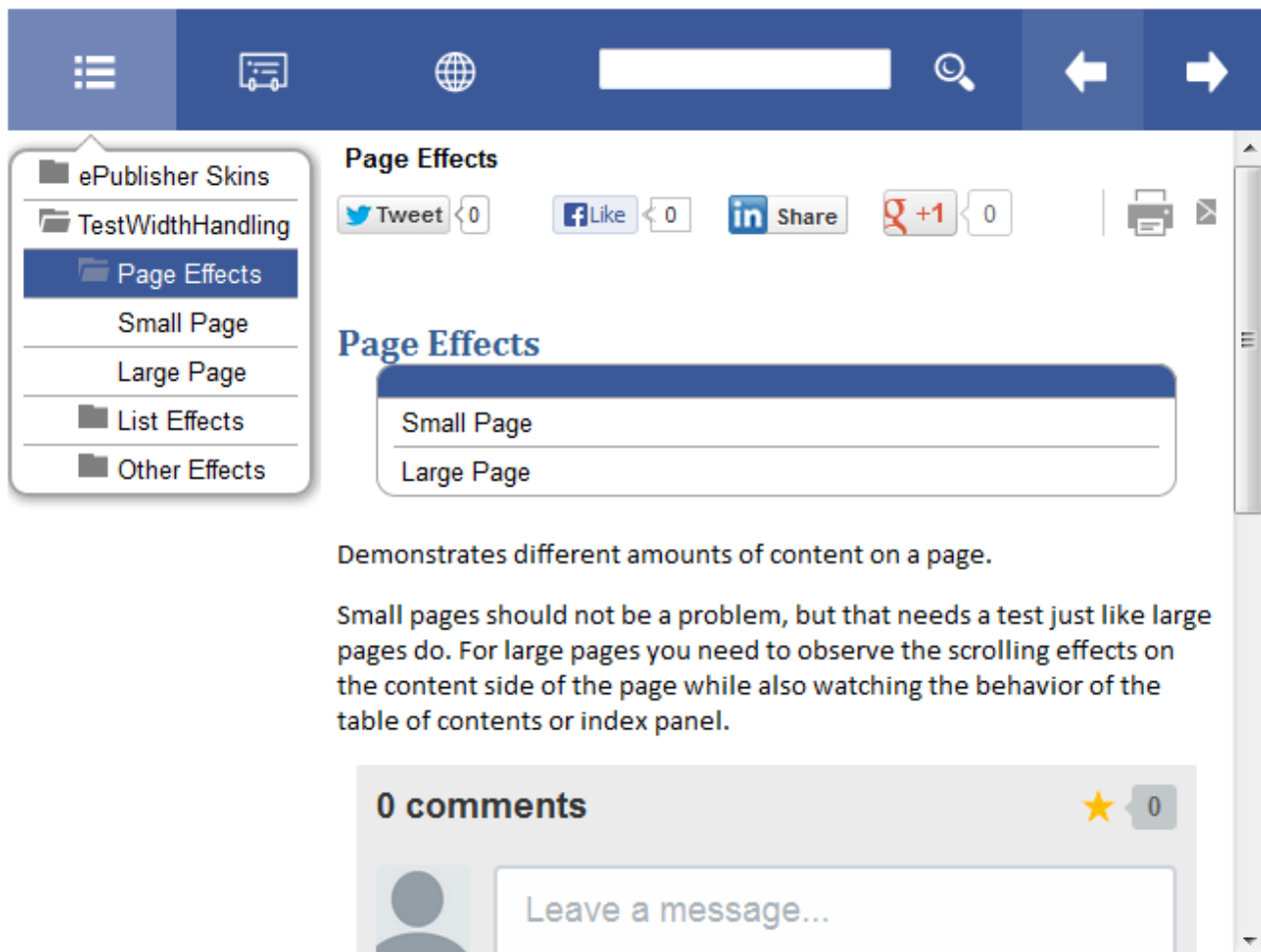
Metro Skin: Color Schemes	
Blue	Orange
Dark Green	Purple
Grey	Red
Light Green	Tiled



Social

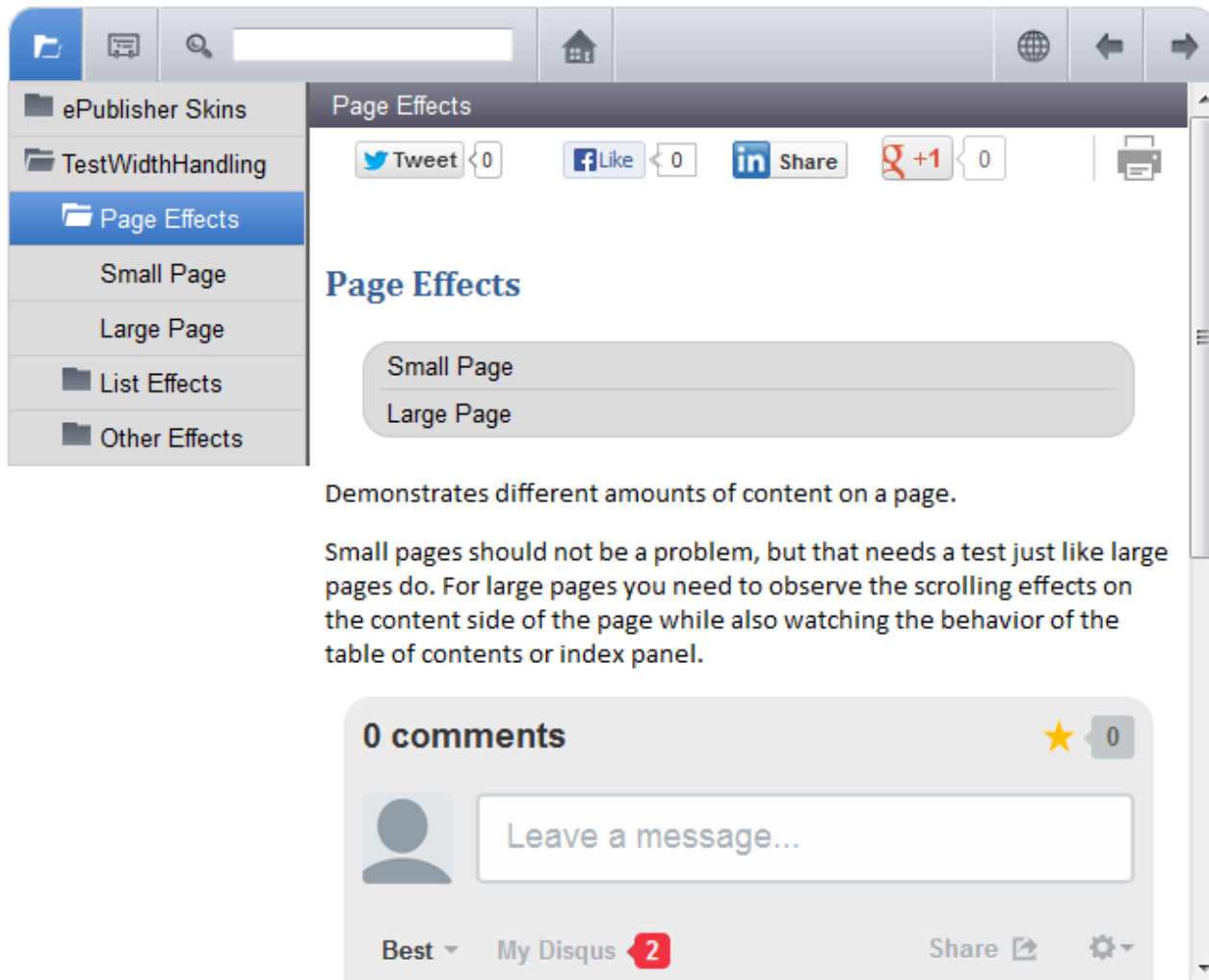
The *Social* skin has been designed to provide a user experience most similar to that of social networking websites. It provides a familiar interface that is intuitive and casual enough to maximize your end-user participation.

Social Skin: Color Schemes
Blue
Green
Grey
Light Blue
Red



Choosing the Compact version of a WebWorks Reverb Skin

For each alternate WebWorks Reverb skin type and color scheme variation, there is also a *Compact* version of that skin. The *Compact* version of the skin is identical in every way except that it uses a smaller size set of toolbar buttons. *Compact* skin types are useful when you know your audience will not likely be using touch devices and thus will not need the larger buttons. The *Compact* skins may seem more acceptable to users of traditional tri-pane help systems on desktop devices. The following is an example of the **Classic - Compact** skin.



Complete List of WebWorks Reverb Skins

The following choices of skins are available with the current installation of ePublisher Designer:

Skin Types	Color Schemes
Classic Classic - Compact	
Corporate Corporate - Compact	Blue Grey Red
Metro Metro - Compact	Blue Dark Green Grey Light Green Orange Purple Red Tiled
Social Social - Compact	Blue Green Grey Light Blue Red

Previewing WebWorks Reverb

You can view the generated files directly in your browser, however, the social-media functionality will not be present when viewing WebWorks Reverb files in this manner. In order to preview the social-media functionality, you will need to view the files from a web server. With ePublisher you can preview your WebWorks Reverb files through a web server without having to configure a separate web server.

To preview the output, simply select the top level group in the **Document Manager**. This will display an entry in the **Output Explorer** called: `View Output`. Double-click on this entry and your help system will be opened in your default browser connected to a web server built into ePublisher.

To fully preview WebWorks Reverb using the built-in web server

1. Open your project and make sure that your WebWorks Reverb target is active and fully generated.
2. In the **Document Manager** select the top level group.
3. In the **Output Explorer** underneath **Merge Output**, double click **View Output**.
4. A browser window will be opened using your default browser and a URL to a `localhost` web address will be displayed. You can now browse the entire help volume as if it were deployed on a dedicated web server.

Delivering WebWorks Reverb

When you generate output for your project, ePublisher creates the `Output` folder in your project folder, and then creates a folder in the `Output` folder for each generated target. Then, ePublisher creates a folder named for the project itself in the target folder. For example, when you generate output, ePublisher creates the following folder structure:

```
component Projects\projectname\Output\targetname\projectname
```

In this folder structure, *component* is the name of the ePublisher component you are using, such as ePublisher Express, *projectname* is the name of your ePublisher project, and *targetname* is the name of your ePublisher target, such as WebWorks Reverb. To deliver your WebWorks Reverb generated output, you need to deliver all the files and subfolders in the *targetname\projectname* folder.

The *targetname\projectname* folder contains the entry-point file, `index.html` by default, which establishes the help set appearance. When the user opens the entry-point file, the browser uses all the files in the *targetname\projectname* folder to display the help, including all the topic files, generated `.css` files, `.pdf` files, images, and WebWorks Reverb components.

Searching WebWorks Reverb - Microsoft IIS Search

If you are not using Client-Side Search with your WebWorks Reverb help system, but instead are using the Microsoft IIS Search option, then use the following guide to assist users in finding terms in your help system:

Boolean

All search words and phrases have an implicit AND

Explicit Boolean Operators

You can use the boolean operators: AND, OR, and NOT. These operators can be used in combination, however they cannot be grouped.

Word Search

For example, searching: `eggs bacon` returns all documents containing “eggs” and “bacon”

Phrase Search

For example, searching “`Good Morning`” returns all documents containing “good” and “morning” where the two words are adjacent

Searching WebWorks Reverb - Client-side Search

If you are not using Microsoft IIS Search with your WebWorks Reverb help system, but instead are using the Client-side Search option, then use the following guide to assist users in finding terms in your help system:

Boolean

All search words and phrases have an implicit AND

Word Search

For example, searching: `eggs bacon` returns all documents containing “eggs” and “bacon”

Phrase Search

For example, searching “`Good Morning`” returns all documents containing “good” and “morning”.

However, the two words will not necessarily be adjacent. In other words, at this time, phrase search returns the same results as a multi-word search.

Searching WebWorks Reverb - URL Method

Search actions can be initiated via URL. For example you can go to the link here:

<http://www.webworks.com/Documentation/Reverb/#search/ePublisher>

When clicked, the link above searches for the term “ePublisher”. This can be changed to any search term, which will produce results accordingly.

Indexing WebWorks Reverb - Client-Side Search for Source Documents, Baggage Files and External URLs

Reverb now has the ability to index baggage files for use in Reverb's Client-side Search. An indexable **Baggage File** in this context is any PDF or HTML file that is linked from a source document that will be included in the generated output for producing useful search results. For more detailed information on baggage files, see "Understanding Baggage Files".

Note: In order to determine what baggage files are indexed, ePublisher examines the file extension and if it matches one on the following then it will be indexed.

- .pdf
- .html
- .htm
- .shtml
- .shtm
- .xhtml
- .xhtm

Baggage files are indexed in the same way that source documents are, as long as the "Client-side Search" is ON (see "Client-side Search"). Indexable baggage files will be indexed as long as the **Index baggage files** setting is **Enabled**. External URLs will be indexed as long as the **Index external links** setting is **Enabled**.

Using Tidy for Indexing HTML Pages

In order to index HTML baggage files, Reverb creates an XHTML copy of the files using Tidy (tool for cleaning up HTML files) to get valid XML files that ePublisher can read. But there might be cases it fails and you will have to teach Tidy how to handle it.

One of the things you might need to teach Tidy about is new tags. You'll know you have to do that if you receive a warning in the log saying something like the following:

```
line 33 column 3 - Error: <not_recognized_tag> is not recognized!
```

When Tidy shows this warning that means it wasn't able to generate an XHTML copy, and therefore ePublisher won't index that baggage file. But fortunately there is a way we can fix that.

To Teach Tidy About New Tags

1. Go to your Tidy directory under the installation directory in your local computer: `...\WebWorks\epublisher\<VERSION>\Helpers\tidy\`
2. Create a Format override of this helper. To do this: in the sub-folder of your project called: `Formats`, where the Format overrides live, create a new folder called `Helpers` and copy the entire folder called `tidy` (from step 1) to this new folder.
3. In the newly created `tidy` folder, open your `config.txt` file.
4. Depending on the kind of tag you want to add, you'll have to uncomment line 8 or 10, or maybe both in the `config.txt` file.
5. Substitute the placeholder we put there and after the colon, with your new tag name (for example: `not_recognized_tag`).
6. Save and close the file.

To know more about how to customize Tidy go to <https://www.w3.org/People/Raggett/tidy/>.

Assigning Relevance Weight to Your Source Documents Styles

Search results are displayed in the Search tab when a user types a word to search for and clicks **Go**. The search results are sorted by the relevancy ranking, which, in case of source documents, is calculated based on the **Search relevance weight** option defined in your **Paragraph** and **Marker Styles**. By default, WebWorks Reverb assigns relevance weight of 1 to all styles.

To Modify the Relevancy Ranking in Source Documents for Search Results

1. Open your project with ePublisher Designer.
2. Scan the document, to pull all styles into the Style Designer.
3. Open the **Style Designer** (F10 or **View > Style Designer**).
4. Select the style you want to assign a weight to (either in **Paragraph Styles** or **Marker Styles**).
5. Open the **Options** window.
6. Change the **Value** of the **Search relevance weight** option to a decimal number you determine or you can just ignore it (which is going to be 0), meaning that the style is not going to be shown in your results.

Assigning Relevance Weight to Your HTML and PDF Baggage Files

The search results are sorted by relevancy ranking, which, in case of HTML baggage files, is calculated based on the scoring preference defined for the HTML tags in the `search_settings.xml` file. By default, WebWorks Reverb assigns relevancy rankings based on where in a topic a particular item is found.

To Modify the Relevancy Ranking in Baggage Files for Search Results

1. Open your project with ePublisher Designer.
2. *If you want to override the relevancy ranking for all WebWorks Help targets*, create the `Formats \WebWorks Reverb\Transforms` folder in your `projectname` folder, where `projectname` is the name of your ePublisher project.
3. *If you want to override the relevancy ranking for one WebWorks Help target*, create the `Targets \WebWorks Help 5.0\Transforms` folder in your `projectname` folder, where `projectname` is the name of your ePublisher project.
4. Create a customization of your `search_settings.xml` file.
5. You'll see the following block of code:

```
<Settings version="1.0" xmlns="urn:WebWorks-Settings-Schema">
<ScoringPrefs default-weight="0.05" pdf-weight="0.05">
  <meta name="keywords" weight="1.0"/>
  <meta name="description" weight="1.0"/>
  <meta name="summary" weight="1.0"/>
  <title weight="1.0"/>
  <div class="myclass" weight="0.05"/>
  <div weight="0.05"/>
  <h1 weight="0.1"/>
  <h2 weight="0.1"/>
  <caption weight="0.1"/>
  <h3 weight="0.1"/>
  <th weight="0.1"/>
  <h4 weight="0.1"/>
  <h5 weight="0.1"/>
  <h6 weight="0.1"/>
  <h7 weight="0.1"/>
  <p weight="0.05"/>
</ScoringPrefs>
</Settings>
```

6. Modify the `weight` attributes for any tags, such as `h1` and `h2`, you want to change. You can also specify additional tags with or without class attributes to further refine weights for your HTML baggage files. You may use decimal values to modify the `weight` attribute value.

Note: If you wish to set a default weight to tags that are not defined in this file simply update the `default-weight` attribute value.

Note: You can change the default weight for all of the text in a PDF file by changing the `pdf-weight` attribute value.

7. Save and close the `search_settings.xml` file.
8. Regenerate your project to review the changes.

Search Highlighting in Baggage Files - Client-Side Search

When you click on a result in your Search Results, you'll open the associated source document or baggage file. If what you are clicking is a baggage file and you want to get the highlighting feature in your baggage file, you'll have to copy next to your file and then reference the `reverb-search.js` script in the `<head>` tag of your HTML file. The `reverb-search.js` file lives in the installation directory at `...\WebWorks\ePublisher\<VERSION>\Formats\WebWorks Reverb\API\reverb-search.js`.

To Reference the `reverb-search.js` File From Within Your HTML File (After Copying the Script Next to Your HTML File)

1. Open your HTML document.
2. Locate the `<head>` tag.
3. Create the following line inside the `<head>` tag, pointing to the script you just copied:

```
<script type="text/javascript" src="../../../reverb-search.js"></script>
```
4. Save your HTML file.
5. You can either Enable in your **Target Settings** under **Baggage Files** the **Copy baggage file dependents** (this will copy the script to the Output folder, see “Copy baggage file dependents”), or you can manually copy the file to the Output directory, next to your baggage file.

URL Toggle in WebWorks Reverb

You can toggle between the Table of Contents or the Index in the Reverb output. For example, you can go to:

<http://www.webworks.com/Documentation/Reverb/#toc/>

http://www.webworks.com/Documentation/DITA_1.2_Specification/#index/

Clicking on those links brings up that part of the help by default as opposed to the regular view.

End-user requirements in WebWorks Reverb

End-users must have cookies and JavaScript enabled in their browser. If not the user will be prompted with a localized message asking them to enable JavaScript or cookies to view the content.

It is possible to customize these messages by overriding the proper Page Templates (.asp).

WebWorks Reverb 2.0

WebWorks Reverb 2.0 has many of the features found in WebWorks Help, and goes one step further by providing high-performance load times combined with optimal handling of the end-user's device resolution. This means that users viewing this type of output on a mobile device will be able to view the files in a fast, light-weight manner that is optimal for lower resolutions and/or touch operated screens. Furthermore, users on a more traditional higher-resolution computer monitor will still have the same advantages offered by a complete online help system. In other words, Reverb adapts to the traits of the device that is doing the reading of the content.

WebWorks Reverb 2.0 also offers a commenting and end-user feedback mechanism using the Disqus commenting and discussion platform. This platform is a leading solution on the Internet for comment handling and has no cost and provides automatic hosting of your end-users comments in a way that is transparent and effective.

Reverb also provides easy Google Analytics integration. Using Google Analytics in your web files will be very easy to configure with this format.

If you have end-users that require localized help files, WebWorks Reverb 2.0 offers a seamless integration with the Google Translate Element. This feature is very powerful and can be used instead of an expensive and burdensome set of translated files and help volumes, which requires a separate set of content for each language that you are supporting. Using WebWorks Reverb 2.0, you can now create instantaneously localized help volumes without any translation requirements and only one set of help files.

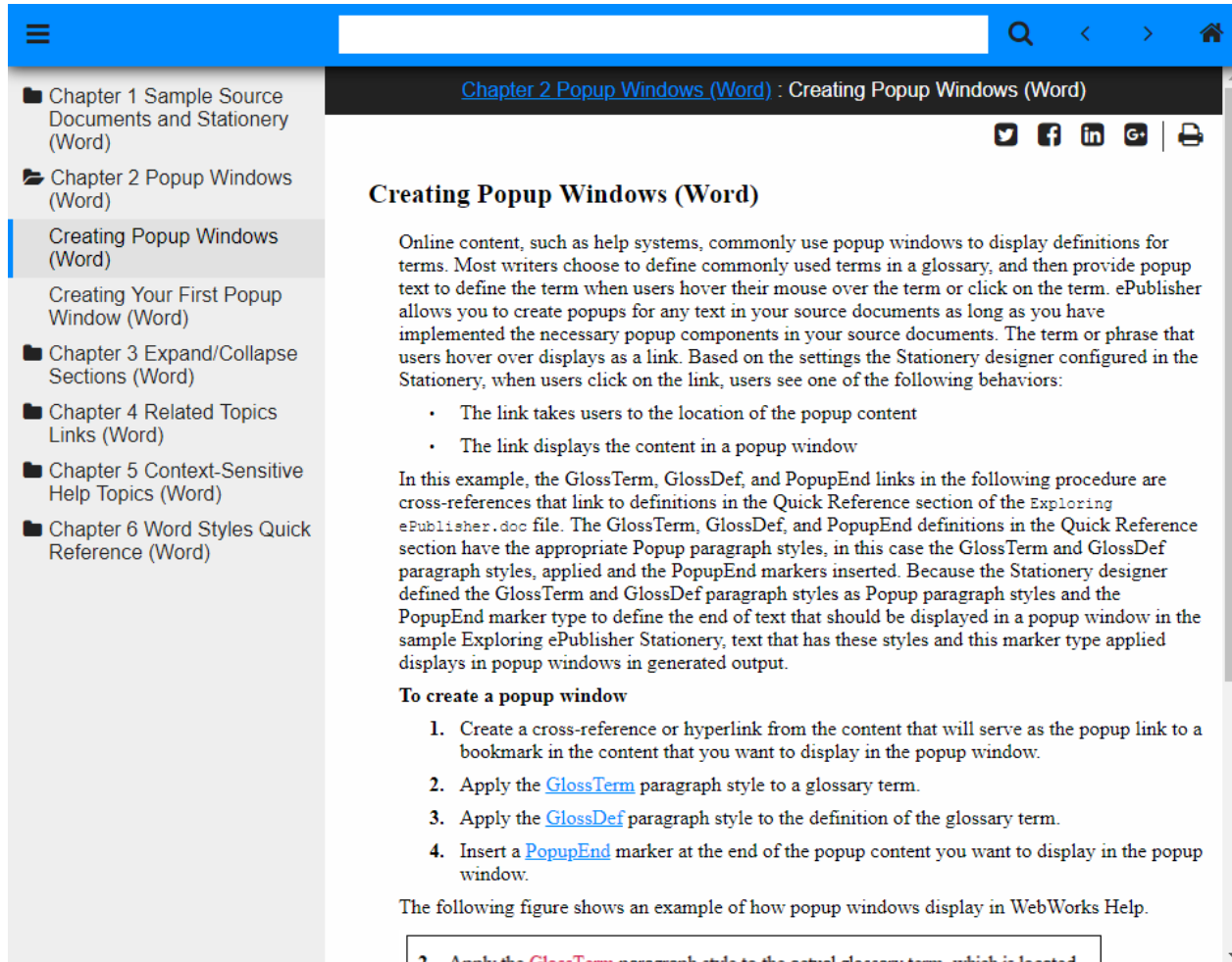
Choosing a Skin in WebWorks Reverb 2.0

You can select from a number of predefined skins that are available for the WebWorks Reverb 2.0 output format. Each skin has been professionally designed with assistance from both graphic designers and web developers so that your users can get an ideal experience when browsing your documentation.

You may select among the following types of skins:

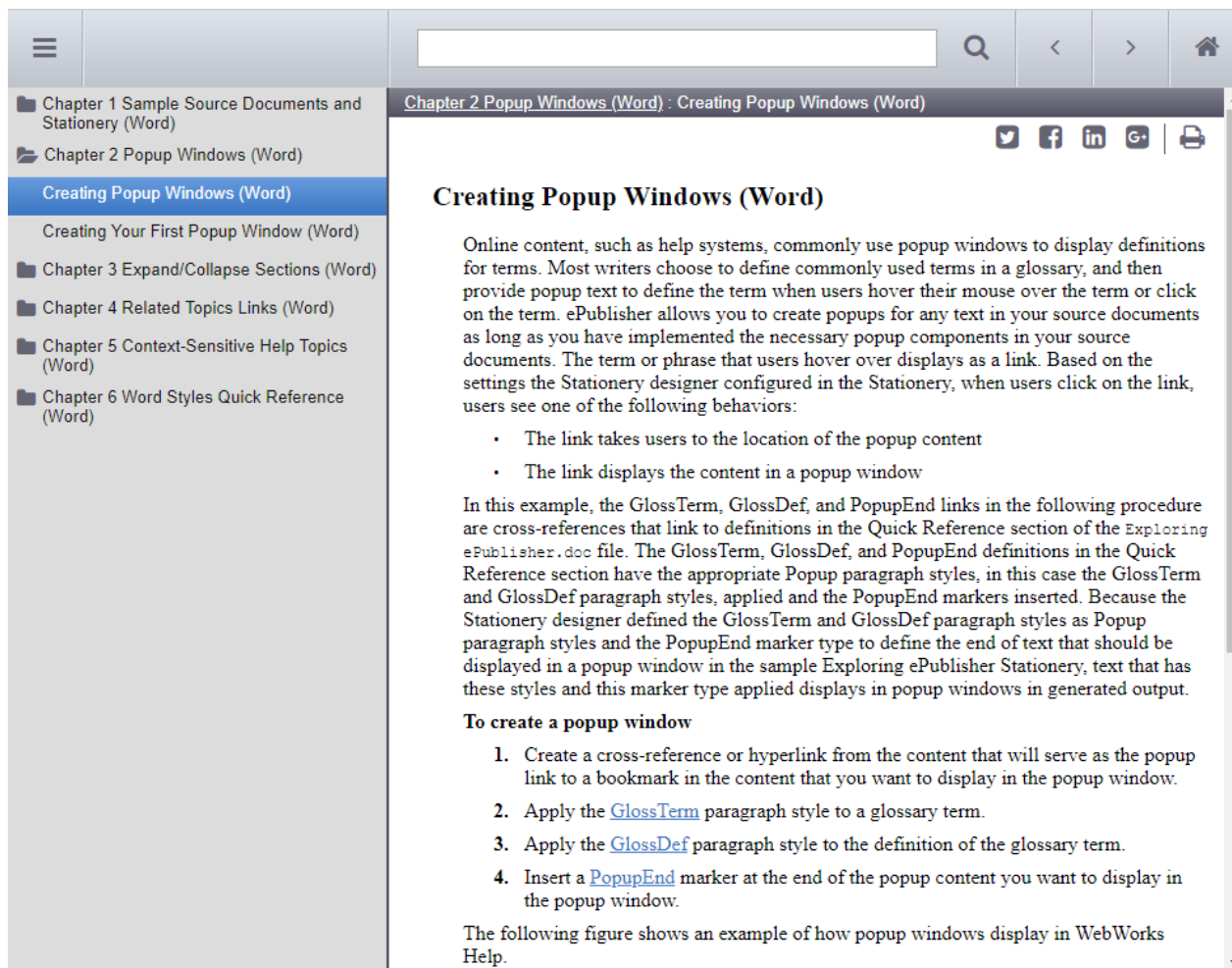
Neo

The default skin for WebWorks Reverb 2.0, Neo is the latest design for Reverb output and was designed with current web aesthetics in mind. It features a simplistic layout that looks great across many devices. This skin is very versatile and well suited for most purposes.



Classic

Based off of the original WebWorks Reverb skin, Classic makes a return with some modifications in WebWorks Reverb 2.0. This skin features a larger toolbar, and gradients across the layout to produce a traditional help look and feel. This skin is great for many purposes and offers existing users a similar skin if they were previously using the Classic skin with WebWorks Reverb.



Corporate

The Corporate skin has been brought into the WebWorks Reverb 2.0 Format, with an updated yet familiar look to it. This skin has a polished look and would suit the needs of a high-profile technology company's help set.

Chapter 1 Sample Source Documents and Stationery (Word) ▶

Chapter 2 Popup Windows (Word) ▼

Creating Popup Windows (Word)

Creating Your First Popup Window (Word)

Chapter 3 Expand/Collapse Sections (Word) ▶

Chapter 4 Related Topics Links (Word) ▶

Chapter 5 Context-Sensitive Help Topics (Word) ▶

Chapter 6 Word Styles Quick Reference (Word) ▶

Chapter 2 Popup Windows (Word) : Creating Popup Windows (Word)

Online content, such as help systems, commonly use popup windows to display definitions for terms. Most writers choose to define commonly used terms in a glossary, and then provide popup text to define the term when users hover their mouse over the term or click on the term. ePublisher allows you to create popups for any text in your source documents as long as you have implemented the necessary popup components in your source documents. The term or phrase that users hover over displays as a link. Based on the settings the Stationery designer configured in the Stationery, when users click on the link, users see one of the following behaviors:

- The link takes users to the location of the popup content
- The link displays the content in a popup window

In this example, the GlossTerm, GlossDef, and PopupEnd links in the following procedure are cross-references that link to definitions in the Quick Reference section of the Exploring ePublisher.doc file. The GlossTerm, GlossDef, and PopupEnd definitions in the Quick Reference section have the appropriate Popup paragraph styles, in this case the GlossTerm and GlossDef paragraph styles, applied and the PopupEnd markers inserted. Because the Stationery designer defined the GlossTerm and GlossDef paragraph styles as Popup paragraph styles and the PopupEnd marker type to define the end of text that should be displayed in a popup window in the sample Exploring ePublisher Stationery, text that has these styles and this marker type applied displays in popup windows in generated output.

To create a popup window

1. Create a cross-reference or hyperlink from the content that will serve as the popup link to a bookmark in the content that you want to display in the popup window.
2. Apply the [GlossTerm](#) paragraph style to a glossary term.
3. Apply the [GlossDef](#) paragraph style to the definition of the glossary term.
4. Insert a [PopupEnd](#) marker at the end of the popup content you want to display in the popup window.

The following figure shows an example of how popup windows display in WebWorks Help.

Metro

The Metro skin has returned in WebWorks Reverb 2.0. This skin was designed in reference to the Windows 8.1 Metro interface design pattern. The Metro skin is useful for users wanting migrate from WebWorks Reverb and have a starting point that looks and feels like their previous Metro skin.

Chapter 2 Popup Windows (Word) : Creating Popup Windows (Word)

Creating Popup Windows (Word)

Online content, such as help systems, commonly use popup windows to display definitions for terms. Most writers choose to define commonly used terms in a glossary, and then provide popup text to define the term when users hover their mouse over the term or click on the term. ePublisher allows you to create popups for any text in your source documents as long as you have implemented the necessary popup components in your source documents. The term or phrase that users hover over displays as a link. Based on the settings the Stationery designer configured in the Stationery, when users click on the link, users see one of the following behaviors:

- The link takes users to the location of the popup content
- The link displays the content in a popup window

In this example, the GlossTerm, GlossDef, and PopupEnd links in the following procedure are cross-references that link to definitions in the Quick Reference section of the Exploring ePublisher.doc file. The GlossTerm, GlossDef, and PopupEnd definitions in the Quick Reference section have the appropriate Popup paragraph styles, in this case the GlossTerm and GlossDef paragraph styles, applied and the PopupEnd markers inserted. Because the Stationery designer defined the GlossTerm and GlossDef paragraph styles as Popup paragraph styles and the PopupEnd marker type to define the end of text that should be displayed in a popup window in the sample Exploring ePublisher Stationery, text that has these styles and this marker type applied displays in popup windows in generated output.

To create a popup window

1. Create a cross-reference or hyperlink from the content that will serve as the popup link to a bookmark in the content that you want to display in the popup window.
2. Apply the [GlossTerm](#) paragraph style to a glossary term.
3. Apply the [GlossDef](#) paragraph style to the definition of the glossary term.
4. Insert a [PopupEnd](#) marker at the end of the popup content you want to display in the popup window.

The following figure shows an example of how popup windows display in WebWorks

Social

The *Social* skin has been designed to provide a user experience most similar to that of social networking websites. It provides a familiar interface that is intuitive and casual enough to maximize your end-user participation.

Chapter 1 Sample Source Documents and Stationery (Word)

Chapter 2 Popup Windows (Word)

Creating Popup Windows (Word)

Creating Your First Popup Window (Word)

Chapter 3 Expand/Collapse Sections (Word)

Chapter 4 Related Topics Links (Word)

Chapter 5 Context-Sensitive Help Topics (Word)

Chapter 6 Word Styles Quick Reference (Word)

Chapter 2 Popup Windows (Word) : Creating Popup Windows (Word)

Creating Popup Windows (Word)

Online content, such as help systems, commonly use popup windows to display definitions for terms. Most writers choose to define commonly used terms in a glossary, and then provide popup text to define the term when users hover their mouse over the term or click on the term. ePublisher allows you to create popups for any text in your source documents as long as you have implemented the necessary popup components in your source documents. The term or phrase that users hover over displays as a link. Based on the settings the Stationery designer configured in the Stationery, when users click on the link, users see one of the following behaviors:

- The link takes users to the location of the popup content
- The link displays the content in a popup window

In this example, the GlossTerm, GlossDef, and PopupEnd links in the following procedure are cross-references that link to definitions in the Quick Reference section of the Exploring ePublisher.doc file. The GlossTerm, GlossDef, and PopupEnd definitions in the Quick Reference section have the appropriate Popup paragraph styles, in this case the GlossTerm and GlossDef paragraph styles, applied and the PopupEnd markers inserted. Because the Stationery designer defined the GlossTerm and GlossDef paragraph styles as Popup paragraph styles and the PopupEnd marker type to define the end of text that should be displayed in a popup window in the sample Exploring ePublisher Stationery, text that has these styles and this marker type applied displays in popup windows in generated output.

To create a popup window

1. Create a cross-reference or hyperlink from the content that will serve as the popup link to a bookmark in the content that you want to display in the popup window.
2. Apply the [GlossTerm](#) paragraph style to a glossary term.
3. Apply the [GlossDef](#) paragraph style to the definition of the glossary term.
4. Insert a [PopupEnd](#) marker at the end of the popup content you want to display in the popup window.

The following figure shows an example of how popup windows display in WebWorks Help.

Using SASS To Customize The WebWorks Reverb 2.0 Interface Design

The WebWorks Reverb 2.0 Format uses a technology called SASS to present the user with a dynamic and responsive visual experience. To summarize, SASS is useful because it lets the designer make use of devices that are typically not available in the CSS language. SASS lets the user create and reuse variables across their style sheet, as well as create functions and mixins to enable efficiency and ease of access that other programming languages typically get to enjoy. Once a SASS design has been created, it is then processed and transcompiled to CSS for use in HTML layouts.

Because of the benefits of this, the WebWorks Reverb 2.0 design process is able to be simple and robust. With a large collection of intuitively-named variables, users are able to change settings like fonts and colors in one spot and watch it propagate throughout the layout.

For those interested, SASS is well documented and WebWorks' implementation of this technology conforms with the standards implemented by the creator. Below are a list of resources that may be useful in priming the user to interacting with SASS for the first time.

SASS Basic Guide: <https://sass-lang.com/guide>

SASS Reference: https://sass-lang.com/documentation/file.SASS_REFERENCE.html

CSS Tutorial: <https://www.w3schools.com/Css/>

For detailed instructions on using SASS to customize WebWorks Reverb 2.0, see “Using SASS To Customize WebWorks Reverb 2.0”

Previewing WebWorks Reverb 2.0

You can view the generated files directly in your browser, however, the social-media functionality will not be present when viewing WebWorks Reverb 2.0 files in this manner. In order to preview the social-media functionality, you will need to view the files from a web server. With ePublisher, you can preview your WebWorks Reverb 2.0 files through a web server without having to configure a separate web server.

To preview the output, simply select the top level group in the **Document Manager**. This will display an entry in the **Output Explorer** called: `View Output`. Double-click on this entry and your help system will be opened in your default browser connected to a web server built into ePublisher.

To fully preview WebWorks Reverb 2.0 using the built-in web server

1. Open your project and make sure that your WebWorks Reverb 2.0 target is active and fully generated.
2. In the **Document Manager** select the top level group.
3. In the **Output Explorer** underneath **Merge Output**, double click **View Output**.
4. A browser window will be opened using your default browser and a URL to a `localhost` web address will be displayed. You can now browse the entire help volume as if it were deployed on a dedicated web server.

Delivering WebWorks Reverb 2.0

When you generate output for your project, ePublisher creates the `Output` folder in your project folder, and then creates a folder in the `Output` folder for each generated target. Then, ePublisher creates a folder named for the project itself in the target folder. For example, when you generate output, ePublisher creates the following folder structure:

```
component Projects\projectname\Output\targetname\projectname
```

In this folder structure, *component* is the name of the ePublisher component you are using, such as ePublisher Express, *projectname* is the name of your ePublisher project, and *targetname* is the name of your ePublisher target, such as WebWorks Reverb. To deliver your WebWorks Reverb generated output, you need to deliver all the files and subfolders in the *targetname\projectname* folder.

The *targetname\projectname* folder contains the entry-point file, `index.html` by default, which establishes the help set appearance. When the user opens the entry-point file, the browser uses all the files in the *targetname\projectname* folder to display the help, including all the topic files, generated `.css` files, `.pdf` files, images, and WebWorks Reverb components.

Understanding Top-Level Groups in WebWorks Reverb 2.0

Top-Level Groups are used for many organizational and functional purposes in WebWorks Reverb 2.0. Once a help set is generated as a WebWorks Reverb 2.0 Output, Top-Level Groups are converted into a hierarchical structure to form the Parcels that make up the Table of Contents, and the structure in the file system. These Parcels can then be selected by the user when using Scoped Search to filter search results, and also using the URL method to filter a help set into a subset that contains a specific section of the entire set.

Searching WebWorks Reverb 2.0- Client-side Search

Use the following guide to assist users in finding terms in your help system:

Boolean

All search words and phrases have an implicit AND

Word Search

For example, searching: `eggs bacon` returns all documents containing “eggs” and “bacon”

Phrase Search

For example, searching “`Good Morning`” returns all documents containing “good” and “morning”.

However, the two words will not necessarily be adjacent. In other words, at this time, phrase search returns the same results as a multi-word search.

Searching WebWorks Reverb - URL Method

Search actions can be initiated via URL. For example you can go to the link here:

<http://www.webworks.com/Documentation/Reverb/index.html#search/ePublisher>

When clicked, the link above searches for the term “ePublisher”. This can be changed to any search term, which will produce results accordingly.

Indexing WebWorks Reverb - Client-Side Search for Source Documents, Baggage Files and External URLs

With WebWorks Reverb 2.0, files can be indexed to produce as search results with the user's help set. An indexable **Baggage File** in this context is any PDF or HTML file that is linked from a source document that will be included in the generated output for producing useful search results. For more detailed information on baggage files, see "Understanding Baggage Files".

Note: In order to determine what baggage files are indexed, ePublisher examines the file extension and if it matches one on the following then it will be indexed.

- .pdf
- .html
- .htm
- .shtml
- .shtm
- .xhtml
- .xhtm

Baggage files are indexed in the same way that source documents are. Indexable baggage files will be indexed as long as the **Index baggage files** Target Setting is **Enabled**. External URLs will be downloaded & indexed as long as the **Index external links** Target Setting is **Enabled**.

Using Tidy for Indexing HTML Pages

In order to index an HTML baggage file, Reverb creates an XHTML copy of the file using Tidy (tool for cleaning up HTML files) to get a valid XML file that ePublisher can read. As useful as Tidy is, there may be times where it does not recognize a tag or generates something improperly. Tidy is configurable and can be adjusted to convert the HTML in the proper way.

When Tidy does not recognize a tag in an HTML file, an error like the following is produced:

```
line 33 column 3 - Error: <not_recognized_tag> is not recognized!
```

This error means that Tidy wasn't able to generate an XHTML copy of the HTML file, and therefore ePublisher won't be able to index it as a baggage file. With the right adjustments, this can be fixed.

Configuring Tidy To Recognize New Tags

1. Go to your Tidy directory under the installation directory in your local computer: `...\WebWorks\ePublisher\<VERSION>\Helpers\tidy\`
2. Create a Format override of this helper. To do this: in the sub-folder of your project called: `Formats`, where the Format overrides live, create a new folder called `Helpers` and copy the entire folder called `tidy` (from step 1) to this new folder.
3. In the newly created `tidy` folder, open your `config.txt` file.
4. Depending on the kind of tag you want to add, you'll have to uncomment line 8 or 10, or maybe both in the `config.txt` file.
5. Substitute the placeholder we put there and after the colon, with your new tag name (for example: `not_recognized_tag`).
6. Save and close the file.

To know more about how to customize Tidy go to <https://www.w3.org/People/Raggett/tidy/>.

Assigning Relevance Weight to Your Source Documents Styles

Search results are displayed in the Search tab when a user types a word to search for. The search results are sorted by a relevancy ranking, which, in the case of source documents, is calculated based on the **Search relevance weight** option defined in your **Paragraph** and **Marker Styles**. By default, WebWorks Reverb 2.0 assigns relevance weight of 1 to all styles.

To Modify the Relevancy Ranking in Source Documents for Search Results

1. Open your project with ePublisher Designer.
2. Scan the document, to pull all styles into the Style Designer.
3. Open the **Style Designer** (F10 or **View > Style Designer**).
4. Select the style you want to assign a weight to (either in **Paragraph Styles** or **Marker Styles**).
5. Open the **Options** window.
6. Change the **Value** of the **Search relevance weight** option to a decimal number you determine or you can just ignore it (which is going to be 0), meaning that the style is not going to be shown in your results.

Assigning Relevance Weight to Your HTML and PDF Baggage Files

The search results are sorted by relevancy ranking, which, in case of HTML baggage files, is calculated based on the scoring preference defined for the HTML tags in the `search_settings.xml` file. By default, WebWorks Reverb 2.0 assigns relevancy rankings based on where in a topic a particular item is found.

To Modify the Relevancy Ranking in Baggage Files for Search Results

1. Open your project with ePublisher Designer.
2. *If you want to override the relevancy ranking for all WebWorks Reverb 2.0 targets*, create the `Formats\WebWorks Reverb 2.0\Transforms` folder in your `projectname` folder, where `projectname` is the name of your ePublisher project.
3. *If you want to override the relevancy ranking for one WebWorks Reverb 2.0 target*, create the `Targets\WebWorks Reverb 2.0\Transforms` folder in your `projectname` folder, where `projectname` is the name of your ePublisher project.
4. Create a customization of your `search_settings.xml` file.
5. You'll see the following block of code:

```
<Settings version="1.0" xmlns="urn:WebWorks-Settings-Schema">
<ScoringPrefs default-weight="0.05" pdf-weight="0.05">
    <meta name="keywords" weight="1.0"/>
    <meta name="description" weight="1.0"/>
    <meta name="summary" weight="1.0"/>
    <title weight="1.0"/>
    <div class="myclass" weight="0.05"/>
    <div weight="0.05"/>
    <h1 weight="0.1"/>
    <h2 weight="0.1"/>
    <caption weight="0.1"/>
    <h3 weight="0.1"/>
    <th weight="0.1"/>
    <h4 weight="0.1"/>
    <h5 weight="0.1"/>
    <h6 weight="0.1"/>
    <h7 weight="0.1"/>
    <p weight="0.05"/>
</ScoringPrefs>
</Settings>
```

6. Modify the `weight` attributes for any tags, such as `h1` and `h2`, you want to change. You can also specify additional tags with or without class attributes to further refine weights for your HTML baggage files. You may use decimal values to modify the `weight` attribute value.

Note: If you wish to set a default weight to tags that are not defined in this file simply update the `default-weight` attribute value.

Note: You can change the default weight for all of the text in a PDF file by changing the `pdf-weight` attribute value.

7. Save and close the `search_settings.xml` file.
8. Regenerate your project to review the changes.

Search Highlighting in Baggage Files - Client-Side Search

When you click on a result in your Search Results, you'll open the associated source document or baggage file. If what you are clicking is a baggage file and you want to get the highlighting feature in your baggage file, you'll have to copy next to your file and then reference the `reverb-search.js` script in the `<head>` tag of your HTML file. The `reverb-search.js` file lives in the installation directory at `...\WebWorks\ePublisher\<VERSION>\Formats\WebWorks Reverb 2.0\API\reverb-search.js`.

To Reference the `reverb-search.js` File From Within Your HTML File (After Copying the Script Next to Your HTML File)

1. Open your HTML document.
2. Locate the `<head>` tag.
3. Create the following line inside the `<head>` tag, pointing to the script you just copied:

```
<script type="text/javascript" src="../../../reverb-search.js"></script>
```
4. Save your HTML file.
5. You can either Enable in your **Target Settings** under **Baggage Files** the **Copy baggage file dependents** (this will copy the script to the Output folder, see “Copy baggage file dependents”), or you can manually copy the file to the Output directory, next to your baggage file.

URL Toggle in WebWorks Reverb

You can toggle between the Table of Contents or the Index in the Reverb output. For example, you can go to:

<http://www.webworks.com/Documentation/Reverb/index.html#toc/>

http://www.webworks.com/Documentation/DITA_1.2_Specification/index.html#index/

Clicking on those links brings up that part of the help by default as opposed to the regular view.

End-user requirements in WebWorks Reverb

End-users must have cookies and JavaScript enabled in their browser. If not, the user will be prompted with a localized message asking them to enable JavaScript or cookies to view the content.

End-users must also consume a WebWorks Reverb 2.0 help set with a browser that is not prior to Internet Explorer 11. Modern browsers are recommended for the best possible experience. If an end-user tries to access a WebWorks Reverb 2.0 help set with an unsupported browser, they will be presented with a message instructing them that the browser they are using is not supported.

It is possible to customize these messages by overriding the proper Page Templates (.asp).

Wiki Markup

As Wikis become more popular and more widely used, the standards continue to evolve. Today, there are several types of Wiki markup. ePublisher allows you to generate Confluence, MediaWiki, and MoinMoin Wiki output. You can also customize this formatting with navigation links, breadcrumbs, and other powerful window elements to meet your requirements. With this output format, you can populate a Wiki with your content and allow your contributors to expand and enhance that content. This output format also provides the navigation elements rarely found in default Wiki implementations.

XML+XSL

The XML+XSL format generates Extensible Markup Language (XML) data files that are formatted by an Extensible Stylesheet Language (XSL) style sheet.

XML+XSL is recommended to create template rules that output specific data and perform other functions, such as sorting data and outputting data only when certain conditions exist in the XML document. You can also repurpose the data for different delivery purposes and mechanisms by customizing the base XSL style sheet. For example, through XSL you can deliver XML or HTML to display in a Web browser, or you can deliver text to display on a wireless phone.

4

Designing Input Format Standards

ePublisher supports various input formats, such as Adobe FrameMaker, Microsoft Word, and DITA. ePublisher provides a comprehensive solution to process your current source documents without any changes in them, based on your existing templates and formats. You can also insert a few additional styles, markers, or field codes in your source documents to implement some online features, such as expand/collapse sections in your generated output.

This section provides many strategies and best practices for preparing your input format standards. If you are designing new templates or reworking your existing standards, review these sections to identify ways to improve your standards. These sections may also give you ideas of ways to reduce maintenance costs for your content in the future. For example, if you have both FrameMaker and Word source documents, you can use the same paragraph style and format names in both source document types so you can more easily manage the styles in your Stationery.

Designing Adobe FrameMaker Formats and Standards

Adobe FrameMaker provides a comprehensive publishing solution with XML-based structured authoring. You can develop the templates you need to deliver polished technical documentation for large product libraries. FrameMaker allows you to create both structured and unstructured content. You can also create DITA-compliant content.

This section describes the design considerations for a FrameMaker catalog and template files. By effectively designing your FrameMaker template, and by consistently applying formats throughout your source documents, you can streamline single-sourcing processes and reduce your production and maintenance costs. This section does not describe all FrameMaker processes, but it focuses on the design considerations related to ePublisher.

Creating Standards to Support Single-Sourcing

To define your FrameMaker standards, create a FrameMaker file with all the elements you need in it, including formats, markers, conditions, variables, tables, and master pages. You can use this file to import and update your standards. For example, you can use this file to update variables and conditions across your FrameMaker source files. You can also use this file as a source document in your Stationery design project. To create a template file, start with one of the default templates provided with FrameMaker that most closely matches the format you want. Then, customize the default file to meet your specific needs. The following sections describe various template areas and considerations, and how to effectively design your FrameMaker template file to support single-sourcing with ePublisher.

Planning for Importing Elements Across Files

You need to carefully plan your FrameMaker standards so you can import all elements, such as formats, page layouts, variables, and conditions across all source files. To avoid issues, do not reuse an element for two different purposes in two different files. For example, if the footer text differs between the front matter and the main chapters of a book, do not use the same variable to define the footer in both files. Otherwise, you cannot import that variable across files.

To avoid conflicts when importing master pages, use different page layout names for special pages in all files, such as Title, TOC right, TOC left, and Index first. In addition, delete unneeded formats, variables, and conditions to simplify your template use and maintenance.

Paragraph Formats in FrameMaker

Create paragraph formats for items based on function, not based on formatting. This approach allows you to modify formatting over time and the format names continue to apply. It also prepares you for structured writing in the future. If you are using DITA, paragraph formats are already defined.

Name your paragraph formats starting with naming conventions that group formats by function. For example, group procedure-related formats together by starting the format names with `Procedure`, such as `ProcedureIntro`, `ProcedureStep1`, `ProcedureStep`, `ProcedureSubStep1`, and `ProcedureSubstep`. You do not need to restart numbering using a `step1` format. If you have a format that always proceeds a numbered list, such as a `ProcedureIntro`, you can restart the numbering with that format, which allows you to not use a `step1`. Either method is fine, but one can require less maintenance when updating steps in a procedure topic.

Note: Format names should not include a period in their name. The period can cause display issues when ePublisher creates the cascading style sheet entry that defines the appearance of the format.

To simplify formatting and save time for future maintenance and customization, set the default paragraph font for all formats, then customize specific formats that need customization. You may need multiple paragraph formats to define functions that support pagination settings, such as a `BodyListIntro` format that has **Keep with next paragraph** set.

In ePublisher, you can scan the source documents to list all the paragraph formats. Then, you can organize them in ePublisher to allow property inheritance and to streamline the customization process for your generated output.

To automate and simplify template use, define the paragraph format that follows each paragraph format. This process allows the writer to press Enter after writing a paragraph and the template creates the next paragraph with the format most commonly used next. For example, after a `Heading` format, the writer most often writes a body paragraph of content.

Common paragraph formats include:

- Anchors for images and tables. You may need multiple indents, such as `Anchor`, `AnchorInList`, and `AnchorInList2`.
- Body paragraphs. You may need multiple indents, such as `Body`, `BodyInList`, and `BodyInList2`.
- Headings, such as `ChapterTitle`, `AppendixTitle`, `Heading1`, `Heading2`, `Heading3`, and `Heading4`. You may also need specialized headings, such as `Title`, `Subtitle`, `FrontMatterHeading1`, `FrontMatterHeading2`, and `FrontMatterHeading3`.
- Bulleted lists. You may need multiple bullet levels, such as `Bullet`, `Bullet2`, `Bullet3`. You may also need a bullet item within a procedure, such as a `ProcedureBullet` and a bullet item within a table, such as a `CellBullet`. For more information, see “Bulleted and Numbered Lists in FrameMaker”.
- Numbered lists. You may need multiple levels, such as `ProcedureStep` that uses numbers and `ProcedureSubstep` that uses lowercase letters. You can use `ProcedureStep1` and `ProcedureSubstep1` to restart numbering, or you can use a common paragraph that precedes each list to restart numbering. You may also need numbered list items in tables, such as `CellStep` and `CellStep1`. Be sure to consider related supporting formats, such as `ProcedureIntro`. For more information, see “Bulleted and Numbered Lists in FrameMaker”.
- Examples, such as code or command syntax statements, usually in a fixed font. To keep the lines of a code example together, you can set the `Example` format to keep with next paragraph and use an `ExampleLast` format to identify the end of the example. You may also need multiple example levels, such as `ExampleInList` and `ExampleInListLast`.
- Paragraphs in tables, such as `CellHeading`, `CellBody`, `CellBody2`, `CellStep`, `CellStep1`, and `CellBullet`. Although you can reuse paragraph formats in tables and adjust the margins when those formats are in a

table in FrameMaker, create unique paragraph formats for use in tables to give you complete control in ePublisher.

- Legal notice and copyright or trademark formats for inside the cover page.
- Table of contents and Index formats. However, these formats are defined on the reference pages rather than as paragraph formats.
- Definition lists, such as term and definition or description. You can use a two-column table for this purpose, but a definition list allows long terms, such as field labels in a user interface, to run across the page without wrapping. Then, the definition or description are indented below the term.
- Header and footer formats to control formatting.
- Notes, cautions, tips, and warnings. You can use the numbering property of a paragraph format to insert default text at the beginning of a paragraph, such as Note, Caution, Tip, or Warning. In ePublisher, you can use the **Bullet** properties for the paragraph style to add an image to the left of each note, caution, tip, or warning.
- Page breaks, which can be identified with a small-font paragraph format with **Keep with previous paragraph** set and a large space below the paragraph that pushes the next paragraph to the next page. This paragraph format is hidden in online content. This approach allows you to put page breaks in the content where needed to achieve the cleanest printed output without customizing the pagination settings for individual paragraphs. However, you need to review all these paragraphs each release and remove unneeded PageBreak paragraphs. This approach increases maintenance, but it prevents format customization for pagination.

ePublisher projects use custom marker types, paragraph formats, and character formats to define online features. You need to give the list of markers and formats to the writers so they know how to implement each online feature. The writers use the markers and formats you create to define online features.

The Stationery defines the custom markers and formats. To reduce complexity, you can use the format names defined in the documentation, or you can define the online feature to a different format. The following list identifies additional paragraph formats you may need to support ePublisher online content features:

- Paragraph or character formats to support multiple languages, such as bidirectional languages and text.
- Dropdown paragraph format that identifies the start of an expand/collapse section. You can end the section with a paragraph format defined to end the section, or with a DropDownEnd marker.
- Popup paragraph formats that define several aspects of popup window content:
 - Popup paragraph format identifies the content to display in a popup window and in a standard help topic. This format is applied to the first paragraph of popup content.
 - Popup Append paragraph format identifies the content to display in a popup window and in a standard help topic. This format is applied to additional popup paragraphs when you have more than one paragraph of content to include in a popup window.
 - Popup Only paragraph format identifies the content to display only in a popup window. This format is applied to the first paragraph of popup content.
 - Popup Only Append paragraph format identifies the content to display only in a popup window. This format is applied to additional popup paragraphs when you have more than one paragraph of content to include in a popup window.
- Related topics paragraph format that identifies a link to a related topic, such as a concept topic related to a task or a task related to a concept.
- See Also paragraph format that identifies the text you want to include in an inline See Also link.

For more information about enabling a specific online feature, see “Designing, Deploying, and Managing Stationery”.

Character Formats in FrameMaker

Create character formats for items based on function, not based on formatting or appearance. This approach allows you to modify formatting over time and the format names continue to apply. It also prepares you for structured writing in the future. If you are using DITA, character formats are already defined.

For character formats, use As Is to start with as base, which allows you to apply multiple character formats to the same text. It also allows each character format to define only the aspects of the formatting required for that character format. Customize each format for your specific need by specifying only the properties required for that format.

Common character formats include:

- Book titles in cross references
- Emphasized text
- Command names
- File and folder names
- User interface items
- Optional steps or if clauses used to introduce optional steps
- Links
- New terms
- Step numbers, which allows you to apply formatting to the number for a step
- Text the user must type
- Variables

ePublisher projects use custom marker types, paragraph formats, and character formats to define online features. You need to give the list of marker types and formats to the writers so they know how to implement each online feature. The writers use the markers and formats you create to define online features.

The Stationery defines the custom marker types, paragraph formats, and character formats. To reduce complexity, you can use the format names defined in the documentation, or you can define the online feature to a different format. The following list identifies additional character formats you may need to support ePublisher online content features:

- Link character format, which identifies the text to include in the link. Include the marker and text in the Link character format.
- Multiple language support, such as bidirectional languages and text, can require a paragraph or character format with Bidi support enabled.
- Abbreviation character format identifies abbreviation alternate text for browsers to display for abbreviations, such as SS#, when a user hovers over the abbreviation in output. Screen readers also can read the abbreviation alternate text. This character format is used in combination with the AbbreviationTitle marker type.
- Acronym character format identifies acronym alternate text for browsers to display for acronyms, such as HTML, when a user hovers over the acronym in output. Screen readers can also read the acronym alternate text. This character format is used in combination with the AcronymTitle marker type.

- Citation character format identifies the source of a quote using a fully-qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. This character format is used in combination with the Citation marker type.
- See Also character format identifies the text you want to include in a See Also button. This format controls the appearance of the text on the button.

For more information about enabling a specific online feature, see “Designing, Deploying, and Managing Stationery”.

Bulleted and Numbered Lists in FrameMaker

ePublisher uses a table-like structure with two columns to display any paragraph format with a hanging indent, such as bulleted and numbered list items, in generated output. ePublisher uses the numbers, characters, formats, and fonts from the source documents for the bullets or numbers. Since some fonts are not available on all computers, you should use character formats in ePublisher to override the formatting of the bullets or numbers. You can also use an image in ePublisher for bullets.

For bulleted lists, you may need multiple bullet levels, such as `Bullet`, `Bullet2`, and `Bullet3`. You may also need a format for a bullet within a procedure, such as a `ProcedureBullet`, and a bullet within a table, such as a `CellBullet`. Make sure you consider all supporting formats you may need, such as a `ListIntro` format for the paragraph that introduces the bulleted list, which should be set to stay with the list (Keep with Next).

For numbered lists, you may need multiple levels, such as a `ProcedureStep` that uses numbers and a `ProcedureSubstep` that uses lowercase letters. To restart numbering, you can use a `ProcedureStep1` and a `ProcedureSubstep1` format. If you have a common paragraph format that precedes each list, you can use that paragraph format to restart numbering, which would eliminate the need for a `ProcedureStep1` format. You may also need a numbered list item in tables, such as `CellStep` and `CellStep1`. Make sure you consider all supporting formats you may need, such as a `ProcedureIntro` format.

Image Formats and Considerations in FrameMaker

If ePublisher cannot use an original image in the output, or if ePublisher determines it needs to modify the image based on how it is included in the source document, ePublisher rasterizes the image using the options you define for your graphic styles in Style Designer. For example, you can define the dots per inch (DPI) and format for the final images. Rasterization of an image can cause the image to be less clear in the output.

To avoid reduced image quality in your output, and to avoid an extended transformation time during the Image stage and pipeline, review the following considerations:

- When ePublisher encounters an anchored frame in your FrameMaker source documents, ePublisher checks for the following conditions:
 - Is the frame a different size than the original image?
 - Is there white space in the frame?
 - Is the image copied into the document, rather than imported by reference?
 - Is the original image a file format other than .jpg, .gif, .png, or .svg?
 - Are there additional elements in the frame, such as text boxes, multiple images, or callouts?

If ePublisher determines that any of these conditions apply, ePublisher rasterizes the entire frame and applies the options you defined in Style Designer.

- To display images at full size in online output and avoid resizing, which can cause the image to be rasterized, set the **By reference graphics use document dimensions** option for your graphic styles to **Disabled**.
- If you want ePublisher to rasterize all images according to your Style Designer options, set the **By reference graphics** option to **Disabled** for all graphic styles.
- When ePublisher finds an image included by-reference that is the original size, is shrink-wrapped, and contains no callouts, ePublisher copies the image directly into the output folder in most cases, bypassing the graphic style options.
- To improve the image quality in your output, resize your images as needed using an image editing application before importing them, rather than adjusting the DPI or scale in FrameMaker. Otherwise, an image included by reference retains its original file size, and it is either scaled by the browser or rasterized according to the size of the anchored frame, which can result in a distorted image.
- For the best compatibility with most computer monitors, save and import your images at 96 DPI using a format that ePublisher does not rasterize.
- Image callouts are useful in many publications. However, text boxes and line drawings cause images to be rasterized, which can make images less clear in your output. Add and edit callouts in your image editing application and then import the single, final image to avoid the rasterization process.
- If you use .svg image files, you need to configure the .svg options to specify whether to rasterize these images. Some output formats and some browsers do not support .svg image files.
- You can add text boxes with GraphicStyle markers to your images without causing the image to be rasterized, since markers do not affect the appearance or format of an anchored frame.
- ePublisher does not include images from FrameMaker Master or Reference pages, and it does not include content outside the main text flow.

- Store image files and source documents on the local computer when generating output.

To achieve the best results when inserting images in FrameMaker

1. Create a unique paragraph format for images. Use the paragraph alignment properties to control the position of your images. Make sure the **Fixed** check box in the **Line Spacing** section is *not* selected for the paragraph format.
2. Insert an anchored frame in an empty paragraph of the format created in step 1.
3. Import your image file by reference into the anchored frame rather than copying it into the document. ePublisher supports only .jpg, .gif, .png, and .svg files. ePublisher rasterizes all other formats.
4. Import the image at the native resolution of the image.
5. Shrink-wrap the frame (type Esc+m+p with the frame selected) and change its **Anchoring Position** to **At Insertion Point** or **Below Current Line**.

Table Formats in FrameMaker

Table formats allow you to define standard tables and quickly apply those standards to tables in your source documents. When you define your table formats, be sure to consider the various types of tables you may need, such as with lines, without lines, checklists, and action/result tables. You can use a table without lines to layout content within an area on a page, such as a definition list with short terms. You can also create a table format for each indent position needed. For example, you can create a table format to use for tables within a bulleted list that is indented to align with the text of each bulleted list item.

ePublisher allows you to define how the header, footer, and main rows of a table appear in your generated output. To support these formatting properties, your tables must have each of these parts defined in your source documents. If a table does not have a header defined, ePublisher cannot apply the formatting defined for the header row.

ePublisher applies the paragraph and character formats you define for content within each cell. You can also configure ePublisher to ignore character formats in a table. You may need additional paragraph formats to use in tables, such as CellBody and CellBullet, so you can define the proper margins and appearance for your generated output. You cannot adjust paragraph formats to change their appearance when used within tables.

Cross Reference Formats in FrameMaker

Cross reference formats allow you to quickly use consistent cross references throughout your source documents. However, you probably want to change the appearance of your cross references in your generated output. For example, you may not want to include page numbers in your online content. ePublisher allows you to define how each cross reference format appears in your generated output.

Define the cross reference formats you need in your source documents, such as references to headings, steps, figures, tables, and chapters. Then, you can define each of these formats separately in ePublisher.

Markers in FrameMaker

FrameMaker uses markers to implement standard features, such as index entries and hypertext links. ePublisher recognizes these standard markers and uses them to implement these standard features in your generated output.

ePublisher projects also use custom marker types, paragraph formats, and character formats to define online features. You need to give the list of marker types and formats to the writers so they know how to implement each online feature. The writers use the markers and formats you create to define online features.

The Stationery defines the custom marker types, paragraph formats, and character formats. Markers with reserved names have their functions defined by default. You can use these default names, or you can create your own markers. To reduce complexity, use the default marker names, which are also used throughout the documentation. You can also use the format names defined in the documentation to reduce complexity. The following table lists the default custom marker types used to implement online features.

Marker Type	Description
AbbreviationTitle	Specifies abbreviation alternate text for browsers to display for abbreviations such as SS# when a user hovers over the abbreviation in output. Screen readers also can read the abbreviation alternate text. Used in combination with the Abbreviation character format.
AcronymTitle	Specifies acronym alternate text for browsers to display for acronyms such as HTML when a user hovers over the acronym in output. Screen readers can also read the acronym alternate text. Used in combination with the Acronym character format.
Citation	Specifies the source of a quote using a fully qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. Used in combination with the Citation character format.
Context Plugin	Specifies context plug-ins for Eclipse help systems. Other Eclipse plug-ins can use the context plug-in IDs to call the Eclipse help system. For more information, see “Using Markers to Specify Context Plug-ins in Eclipse Help”.
DropDownEnd	Marks the end of an expand/collapse section. Used in conjunction with an Expand/Collapse paragraph format.
Filename	Specifies the name of an output file for a page or an image.
GraphicScale	Specifies a percentage to use to resize an image, such as 50 or 75 percent, in generated output.
GraphicStyle	Specifies the name of a graphic style defined in a project to apply to an image. This marker type is an internal marker type that is not displayed in Stationery Designer. You cannot create a marker type with a different name and assign it this functionality.
Hypertext	Specifies a link using the <code>newlink</code> and <code>gotolink</code> commands in Adobe FrameMaker. This marker type is a default Adobe FrameMaker marker type ePublisher automatically maps.
ImageAltText	Specifies alternate text for an image. This text is added to the <code>alt</code> attribute of the <code>img</code> tag in the output. Screen readers use this text when you create accessible content.
ImageAreaAltText	Specifies alternate text for clickable regions in an image map. This text is added to the <code>alt</code> attribute of the <code>img</code> tag in the output. Screen readers use this text when you create accessible content.
ImageLongDescByRef	Specifies the path to the file that contains the long description for an image. This text is added to the <code>longdesc</code> attribute of the <code>img</code> tag in the output. Screen readers read this description when you create accessible content.

Marker Type	Description
ImageLongDescNotReq	Specifies that a long description is not required for an image, which bypasses this accessibility check for the image when you create accessible content.
ImageLongDescText	Specifies the long description for an image. This text is added to the <code>longdesc</code> attribute of the <code>img</code> tag in the output. Screen readers read this description when you create accessible content.
Keywords	Specifies the keywords to include in the <code>meta</code> tag for the topic. The <code>meta</code> tag improves searchability on the Web.
PageStyle	Specifies the name of a page style defined in the project to apply to a topic. This marker type is an internal marker type that is not displayed in Stationery Designer. You cannot create a marker type with a different name and assign it this functionality.
PassThrough	Specifies that ePublisher place the contents of the marker directly into the generated output without processing the content in any way. For example, you could use a PassThrough marker if you wanted to embed HTML code within your generated output.
Popup	Specifies the start of the content to include in a popup window. The content is displayed in a popup window when you hover over the link. When you click the link in some output formats, the topic where the popup text is stored, such as the glossary, is displayed.
PopupEnd	Marks the end of the content to include in a popup window.
PopupOnly	Specifies the start of the content to include in only a popup window. Browsers display the content in a popup window when you hover over or click the link.
RubiComposite	No longer supported.
SeeAlsoKeyword	Specifies an internal identifier for a topic. SeeAlsoLink markers in other topics can list this identifier to create a link to this topic. Used in conjunction with a See Also paragraph format or character format.
SeeAlsoLink	Identifies an internal identifier from another topic to include in the list of See Also links in this topic. Used in conjunction with a See Also paragraph format or character format.
SeeAlsoLinkDisplayType	Specifies whether to display the target topics on a popup menu or in a window. By default, the links are displayed in the Topics Found window. To display a popup menu, set the value to <code>menu</code> . This marker type is supported only in HTML Help.
SeeAlsoLinkWindowType	Specifies the name of the window defined in the <code>.hhp</code> file, such as TriPane or Main, that the topic opens in when the user clicks the link. This marker type is supported only in HTML Help.
TableSummary	Specifies an alternate text summary for a table, which is used when you create accessible content. This text is added to the <code>summary</code> attribute of the <code>table</code> tag in the output. Screen readers read this description when you create accessible content.
TableSummaryNotReq	Specifies that a summary is not required for a table, which bypasses this accessibility check for that table.
TOCIconHTMLHelp	Identifies the image to use as the table of contents icon for a topic in the HTML Help output format.
TOCIconJavaHelp	Identifies the image to use as the table of contents icon for a topic in the Sun JavaHelp output format.
TOCIconOracleHelp	Identifies the image to use as the table of contents icon for a topic in the Oracle Help output format.
TOCIconWWHelp	Identifies the image to use as the table of contents icon for a topic in the WebWorks Help output format.

Marker Type	Description
TopicAlias	Specifies an internal identifier for a topic that can be used to create a context-sensitive link to that topic.
TopicDescription	Specifies a topic description for a context-sensitive help topic in Eclipse help systems. For more information, see “Using Markers to Specify Topic Descriptions for Context-Sensitive Help Topics in Eclipse Help”.
WhatIsThisID	Identifies a What Is This help internal identifier for creating context-sensitive What Is This field-level help for Microsoft HTML Help.
WikiCategory	Specifies the category or label you want to assign to a topic when generating Wiki output. For more information, see “Defining Wiki Categories or Labels”.
WindowType	Specifies the name of the window defined in the help project that the topic should be displayed in. In Microsoft HTML Help, the window names are defined in the .hhp file. This marker type is supported in Microsoft HTML Help and Oracle Help.

Variables and Conditions in FrameMaker

Variables allow you to define a phrase once and consistently use that phrase throughout your source documents. Then, if you ever need to change that phrase, you can change it in one location and apply that change throughout your source documents. For example, you can use variables for product names, book titles, and company names.

Conditions allow you to single-source information and include or exclude specific sets of information. You can apply a condition to a character, word, sentence, paragraph, or entire sections of content. Then, you can specify whether to show or hide the content with that condition applied to it. This capability allows you to create multiple version of content based on your specific needs. You can also use conditions to include and exclude notes to the writer or reviewer during the content development process. When you combine variables and conditions, you can customize information for multiple versions of a product while reducing your maintenance costs by reducing duplicate information.

When working with conditions, you can customize the appearance of content with a condition applied by using color, underline, overline, and strikethrough formatting for the condition. This formatting helps you maintain and work with the content. However, ePublisher does not display this formatting in the generated output. If you want to apply formatting in the generated output, use paragraph and character formats to define the appearance for the content.

To simplify consistently setting variables and conditions across your source documents, create a standard file with all the variables and conditions defined in it. The file can display the value of each variable and the show/hide state of each condition. Writers can set the variables and conditions as needed in this one file. Then, the writers can import the variables and conditions from this file into all the source documents.

Note: Use each variable and condition for the same purpose and value in all source documents. For example, if you want the footer in the preface file to be different from the footer in the chapter files, use a different variable to define the footer in each file. Otherwise, you cannot import variables across all the source documents.

Page Layouts in FrameMaker

A **master page** defines the layout of one or more pages and includes all design elements, such as headers, footers, background text, and graphics, for every page that uses that master page. A master page allows you to define the layout of multiple pages in one place, and apply that layout to multiple pages. If you want to adjust the page layout, you need change it only in one place. Each template has at least one master page.

You can create multiple master pages, such as one for odd pages, one for even pages, and one for the first page of each chapter. To simplify page management and being able to import master pages across files, do not redefine a master page to have a different layout in different files. For example, if you want odd pages to have a different footer in the front matter than in the chapters, create a master page for each case, such as ChapterOdd and PrefaceOdd.

You may need to create many master pages for special purposes, such as Title, LegalNotice, PrefaceOdd, PrefaceEven, ChapterFirst, ChapterOdd, ChapterEven, AppendixFirst, AppendixOdd, AppendixEven, IndexFirst, IndexOdd, and IndexEven.

Reference Pages, Table of Contents, and Indexes in FrameMaker

Reference pages define default images, lines, heading levels, and formatting for generated table of contents and index files. You can use reference pages to simplify content creation in your source documents. For more information about reference pages and formatting generated table of contents and indexes in your printed content, see the Adobe FrameMaker documentation and help.

Since graphics and lines defined on the reference pages are not in the main flow, ePublisher cannot include these items in the generated output. For images and lines, use anchored frames in your content to include the images by reference.

Since the appearance of online table of contents and indexes often differ from printed versions, you need to be able to deliver customized table of contents and indexes in your online content. Therefore, ePublisher does not need the table of contents and index formatting defined on the reference pages. ePublisher allows you to define the table of contents levels and appearance, as well as the appearance of the index in your generated output. ePublisher uses the index markers throughout your source files to build the online index. This powerful support allows you to deliver the online content you require.

Automation and Scripting in FrameMaker

You can use add-on products, such as FrameScript and IXgen, to automate, customize, and extend the default FrameMaker capabilities. This automation can streamline the content authoring process and save the writers valuable time and effort.

Designing Microsoft Word Templates and Standards

Microsoft Word is a powerful authoring tool that allows you to quickly create professional content. You can develop templates with the styles and other standard elements you need to deliver polished technical documentation. Microsoft Word provides many automation features to streamline the content development process. The new XML-based formats allow Microsoft Word to integrate content with other Microsoft Office products.

This section describes the design considerations for a Microsoft Word template. By effectively designing your Microsoft Word template, and by consistently applying styles throughout your source documents, you can streamline single-sourcing processes and reduce your production and maintenance costs. This section does not describe all Microsoft Word processes, but it focuses on the design considerations related to ePublisher.

Word Standards to Support Single-Sourcing

To define your Microsoft Word standards, create a template `.dot` file with all the elements you need in it, including styles, variables, autotext, toolbars, macros, and page layouts. You can use this file to create all new source documents and to update the styles, autotext, toolbars, and macros in existing source documents. You can also use this file to create a source document to include in your Stationery design project.

The following sections describe various template areas and considerations, and how to effectively design your Microsoft Word template to support single-sourcing with ePublisher.

Understanding the Microsoft Word Template File

The template .dot file defines the default styles, autotext, toolbars, and macros for documents attached to that template. You can also define the starting variables, properties, content, sections, and page layouts for new documents created from the template. The existing bulleted and numbered list items in the template file preload the bullet and number gallery in Microsoft Word, which can cause inconsistencies on multiple computers when these items are not properly defined in the template.

When you create a new document based on a template, the contents of the template file, including text, paragraphs, sections, headers, and footers are copied to the new document. In addition, the new document is attached to the template and uses the styles, autotext, toolbars, and macros defined in the template.

You can also attach an existing document to a template. The document can then use the autotext, toolbars, and macros defined in the template. If you select the **Automatically update document styles** option, the style definitions in the template overwrite the style definitions in the document. The existing document content is not changed, including the headers, footers, and page layout. Style modifications on individual paragraphs, such as the Keep with Next setting, are also not changed or reset. You can create a macro to find each style and reset the paragraph to the default style, which removes the modifications.

Creating a Clean Base Template File

When you create custom styles and macros in Microsoft Word, these customizations are often stored in the default `Normal.dot` file on your computer. In addition, when you use Microsoft Word as your email editor in Microsoft Outlook, some other customizations can be stored in the default `Normal.dot` file. When you create a new Microsoft Word template, you want to use a clean `Normal.dot` file as the starting point for your template. Save the template file with a new name, and then customize the template as needed. To simplify your template use and maintenance, delete the unneeded styles, variables, autotext, and macros.

To create a clean base template file

1. On the **Tools** menu in Microsoft Word, click **Options**.
2. On the **File Locations** tab, find where the User Templates are stored.
3. Close Microsoft Word and all other Microsoft Office products.
4. Open the folder where the User Templates are stored and rename the `Normal.dot` file in that location in case you later need any customizations stored in that file.
5. Open Microsoft Word.
6. On the **File** menu, click **Save As**.
7. In **Save as type**, select **Document Template (*.dot)**.
8. Specify the name and location for your new template, and then click **Save**.

You can customize the new template to meet your specific needs. To create a new tab in the Template Selection window in Microsoft Word, you can create a folder within the User Templates location. Then, you can store your new template in that folder.

Paragraph Styles in Word

Create paragraph styles for items based on function, not based on formatting. This approach allows you to modify formatting over time and the style names continue to apply. It also prepares you for structured writing in the future.

Name your paragraph styles starting with naming conventions that group styles by function. For example, group procedure-related styles together by starting the style names with Procedure, such as ProcedureIntro, ProcedureStep, and ProcedureSubstep.

Note: Style names should not include a period in their name. The period can cause display issues when ePublisher creates the cascading style sheet entry that defines the appearance of the style.

To simplify formatting and save time for future maintenance and customization, set the default paragraph font and spacing for a base style, such as Normal. Then, base other styles on this base style to inherit the default formatting settings. This process allows you to quickly modify fonts and spacing across styles by modifying only the base style. You can customize settings for each style as needed. The customized settings are not affected when you modify those settings in the base style. To simplify maintenance for heading styles, which often use a different font than your content styles, you may want to base all heading styles on the Heading 1 style to define the font for all headings.

In ePublisher, you can scan the source documents to list all the paragraph styles. Then, you can organize them in ePublisher to allow property inheritance and to streamline the customization process for your generated output.

You may need multiple paragraph styles to define functions that support pagination settings, such as a BodyListIntro format that has **Keep with next** set. To reduce the number of paragraph styles, you can customize paragraphs to add the **Keep with next** setting as needed. Customizing this setting on a paragraph does not affect the ability for the paragraph to receive the other formatting settings from the style definition.

To automate and simplify template use, define the paragraph style that follows each paragraph style. This process allows the writer to press Enter after writing a paragraph and the template creates the next paragraph with the style most commonly used next. For example, after a Heading style, the writer most often writes a body paragraph of content.

Common paragraph styles include:

- Figure paragraphs. You may need multiple indents, such as Figure, FigureInList, and FigureInList2.
- Body paragraphs. You may need multiple indents, such as Body, BodyInList, and BodyInList2. To reduce training needs, you can use the default style names, such as Body Text, Body Text Indent, and Body Text Indent 2.
- Headings, such as Heading 1, Heading 2, Heading 3, and Heading 4. You may also need specialized headings, such as Title, Subtitle, FrontMatterHeading1, FrontMatterHeading2, and FrontMatterHeading3. The cross-reference feature in Microsoft Word allows you to create cross references to headings that use the default Heading styles named Heading *X*, where *X* is a number. To create cross references to other styles, use bookmarks. Do not paste content at the beginning of a heading. Existing cross references to that heading may include the pasted content when the cross references are updated.
- Bulleted lists. You may need multiple bullet levels, such as Bullet, Bullet2, Bullet3. You may also need a bullet item within a procedure, such as a ProcedureBullet and a bullet item within a table, such as a CellBullet. For more information, see “Bulleted and Numbered Lists in Word”.
- Numbered lists. You may need multiple levels, such as ProcedureStep that uses numbers and ProcedureSubstep that uses lowercase letters. You may also need numbered list items in tables, such as

CellStep. Be sure to consider related supporting formats, such as ProcedureIntro. For more information, see “Bulleted and Numbered Lists in Word”.

- Examples, such as code or command syntax statements, usually in a fixed font. To keep the lines of a code example together, you can set the **Keep with next** setting for the Example style and use an ExampleLast style to identify the end of the example. You may also need multiple example levels, such as ExampleInList and ExampleInListLast.
- Paragraphs in tables, such as CellHeading, CellBody, CellBody2, CellStep, and CellBullet.
- Legal notice and copyright or trademark styles for inside the cover page.
- Table of contents and Index styles.
- Definition lists, such as term and definition or description. You can use a two-column table for this purpose, but a definition list allows long terms, such as field labels in a user interface, to run across the page without wrapping. Then, the definition or description are indented below the term.
- Header and footer styles to control formatting.
- Notes, cautions, tips, and warnings.

ePublisher projects use custom field code markers, paragraph styles, and character styles to define online features. You need to give the list of markers and styles to the writers so they know how to implement each online feature. The writers use the markers and styles you create to define online features.

The Stationery defines the custom markers and styles. To reduce complexity, you can use the style names defined in the documentation, or you can define the online feature to a different style. The following list identifies additional paragraph styles you may need to support ePublisher online content features:

- Paragraph or character styles to support multiple languages, such as bidirectional languages and text.
- Dropdown paragraph style that identifies the start of an expand/collapse section. You can end the section with a paragraph style defined to end the section, or with a DropDownEnd marker.
- Popup paragraph styles that define several aspects of popup window content:
 - Popup paragraph style identifies the content to display in a popup window and in a standard help topic. This style is applied to the first paragraph of popup content.
 - Popup Append paragraph style identifies the content to display in a popup window and in a standard help topic. This style is applied to additional popup paragraphs when you have more than one paragraph of content to include in a popup window.
 - Popup Only paragraph style identifies the content to display only in a popup window. This style is applied to the first paragraph of popup content.
 - Popup Only Append paragraph style identifies the content to display only in a popup window. This style is applied to additional popup paragraphs when you have more than one paragraph of content to include in a popup window.
- Related topics paragraph style that identifies a link to a related topic, such as a concept topic related to a task or a task related to a concept.
- See Also paragraph style that identifies the text you want to include in an inline See Also link.

For more information about enabling a specific online feature, see “Designing, Deploying, and Managing Stationery”.

Character Styles in Word

Create character styles for items based on function, not based on formatting or appearance. This approach allows you to modify formatting over time and the style names continue to apply. It also prepares you for structured writing in the future.

You can apply only one character style to a set of text. If you apply a second character style to that text, it replaces the initial character style. Therefore, you may need more character styles to address all the possible combinations, such as variables in a paragraph and variables in a code sample.

Common character styles include:

- Book titles in cross references
- Emphasized text
- Command names
- File and folder names
- User interface items
- Optional steps or if clauses used to introduce optional steps
- Links
- New terms
- Step numbers, which allows you to apply formatting to the step number defined with a sequence field code
- Text the user must type
- Variables

ePublisher projects use custom field code markers and styles to define online features. You need to give the list of markers and styles to the writers so they know how to implement each online feature. The writers use the markers and styles you create to define online features.

The Stationery defines the custom markers and styles. To reduce complexity, you can use the style names defined in the documentation, or you can define the online feature to a different style. The following list identifies additional character styles you may need to support ePublisher online content features:

- Multiple language support, such as bidirectional languages and text, can require a paragraph or character style with Bidi support enabled.
- Abbreviation character style identifies abbreviation alternate text for browsers to display for abbreviations, such as SS#, when a user hovers over the abbreviation in output. Screen readers also can read the abbreviation alternate text. This character style is used in combination with the AbbreviationTitle marker type.
- Acronym character style identifies acronym alternate text for browsers to display for acronyms, such as HTML, when a user hovers over the acronym in output. Screen readers can also read the acronym alternate text. This character style is used in combination with the AcronymTitle marker type.
- Citation character style identifies the source of a quote using a fully-qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. This character style is used in combination with the Citation marker type.

- See Also character style identifies the text you want to include in a See Also button. This style controls the appearance of the text on the button.

For more information about enabling a specific online feature, see “Designing, Deploying, and Managing Stationery”.

Bulleted and Numbered Lists in Word

ePublisher uses a table-like structure with two columns to display any paragraph style with a hanging indent, such as bulleted and numbered list items, in generated output. ePublisher uses the numbers, characters, and fonts from the source documents for the bullets or numbers. Since some fonts are not available on all computers, you should use character styles in ePublisher to override the formatting of the bullets or numbers. You can also use an image in ePublisher for bullets.

Bulleted and numbered list items can have inconsistent formatting from one computer to another. The formatting is defined by the Bullets and Numbering style gallery. To correctly populate the style gallery, the source document must have an example of each type of bulleted and numbered item. Leave the correct example of each item in the document until that style type is used. If a correctly formatted item does not exist in the document, copy the correct style from the template and paste it in the document to create the first item with that style.

Bulleted Lists in Word

For bulleted lists, you may need multiple bullet levels, such as Bullet, Bullet2, and Bullet3. You may also need a format for a bullet within a procedure, such as a ProcedureBullet, and a bullet within a table, such as a CellBullet. Make sure you consider all supporting formats you may need, such as a ListIntro format for the paragraph that introduces the bulleted list, which should be set to stay with the list (Keep with Next).

Do not base two bulleted items with different bullets on the same base style. Otherwise, when you modify the bullet for one bulleted item style, the bullet on the other item is also affected.

Numbered Lists in Word

For numbered lists, you may need multiple levels, such as a ProcedureStep that uses numbers and a ProcedureSubstep that uses lowercase letters. You may also need a numbered list item in tables, such as CellStep. Make sure you consider all supporting formats you may need, such as a ProcedureIntro format.

Microsoft Word can have issues with the built-in autonumbering when restarting the numbering in lists. To avoid these issues, you can use sequence field codes to completely control numbering in your source documents. You can define a paragraph style with a hanging indent for each numbered step item. Define a character style and apply it to the sequence field code to control how the numbered list appears both in print and in your generated output. Autotext can help you automate list creation. You can also create a macro to quickly number a set of paragraphs. To restart a field code, add the \r1 option.

Image Styles and Considerations in Word

If ePublisher cannot use an original image in the output, or if ePublisher determines it needs to modify the image based on how it is included in the source document, ePublisher rasterizes the image using the options you define for your graphic styles in Style Designer. For example, you can define the dots per inch (DPI) and format for the final images. Rasterization of an image can cause the image to be less clear in the output.

To avoid reduced image quality in your output, and to avoid an extended transformation time during the Image stage and pipeline, review the following considerations:

- When ePublisher encounters an image in your Microsoft Word source documents, ePublisher checks for the following conditions:
 - Is the frame a different size than the original image?
 - Is there white space in the frame with the image?
 - Is the image copied into the document, rather than imported by reference?
 - Is the original image a file format other than .jpg, .gif, .png, or .svg?
 - Are there additional elements in the frame, such as text boxes, multiple images, or callouts?

If ePublisher determines that any of these conditions apply, ePublisher rasterizes the entire frame and applies the options you defined in Style Designer.

- To display images at full size in online output and avoid resizing, which can cause the image to be rasterized, set the **By reference graphics use document dimensions** option for your graphic styles to **Disabled**.
- If you want ePublisher to rasterize all images according to your Style Designer options, set the **By reference graphics** option to **Disabled** for all graphic styles.
- When ePublisher finds an image included by-reference that is the original size and contains no callouts, ePublisher copies the image directly into the output folder in most cases, bypassing the graphic style options.
- To improve the image quality in your output, resize your images as needed using an image editing application before importing them, rather than adjusting the size in Microsoft Word. Otherwise, an image included by reference retains its original file size, and it is either scaled by the browser or rasterized according to the size in the source document, which can result in a distorted image.
- For the best compatibility with most computer monitors, save and import your images at 96 DPI using a format that ePublisher does not rasterize.
- Image callouts are useful in many publications. However, text boxes and line drawings cause images to be rasterized, which can make images less clear in your output. Add and edit callouts in your image editing application and then import the single, final image to avoid the rasterization process.
- You can add text boxes with GraphicStyle markers to your images without causing the image to be rasterized, since markers do not affect the appearance of the image.
- Store image files and source documents on the local computer when generating output.

To achieve the best results when inserting images in Microsoft Word

1. Create a unique paragraph style for images. Use the paragraph alignment properties to control the position of your images.
2. Import your image file by reference onto the unique paragraph rather than copying it into the document. ePublisher supports only .jpg, .gif, .png, and .svg files. ePublisher rasterizes all other formats.
3. Do not resize the image in Microsoft Word.

Table Styles in Word

Table styles, which are available in recent versions of Microsoft Word, allow you to define standard tables and quickly create tables with those standards in your source documents. If your version of Microsoft Word does not support table styles, use the TableStyle marker to specify the style to apply in ePublisher that defines the appearance of the table.

Table styles are often overlooked in Microsoft Word. The default template provides many default table styles, such as Table Grid and Table Normal. You can create custom table styles for your specific requirements. Define table header rows for each table that repeat when the table splits across pages, and do not allow rows to break across pages, which can create awkward breaks within tables in your printed content. You can use autotext to quickly create standard tables in your source documents.

When you define your table styles, be sure to consider the various types of tables you may need, such as with lines, without lines, checklists, and action/result tables. You can use a table without lines to layout content within an area on a page, such as a definition list with short terms. You can also create a table style for each indent position needed. For example, you can create a table style to use for tables within a bulleted list that is indented to align with the text of each bulleted list item.

ePublisher allows you to define how the header, footer, and main rows of a table appear in your generated output. To support these formatting properties, your tables must have each of these parts defined in your source documents. If a table does not have a header defined, ePublisher cannot apply the formatting defined for the header row. Microsoft Word does not support table footers, so the footer formatting settings in ePublisher do not apply to Microsoft Word source documents.

ePublisher applies the paragraph and character styles you define for content within each cell. You can also configure ePublisher to ignore character styles in a table. You may need additional paragraph styles to use in tables, such as CellBody and CellBullet, so you can define the proper margins and appearance for your generated output.

Field Codes

Microsoft Word uses field codes to implement standard features, such as index entries and variables. You can use sequence field codes for numbering, such as numbering chapters, steps, figure captions, and table captions. ePublisher recognizes many of the standard field codes and uses them to implement these standard features in your generated output.

ePublisher projects also use custom field codes (markers) and styles to define online features. The toolbar provided by ePublisher in Microsoft Word allows you to quickly insert the custom markers. You need to give the list of markers and styles to the writers so they know how to implement each online feature. The writers use the markers and styles you create to define online features.

The Stationery defines the custom markers and styles. Markers with reserved names have their functions defined by default. You can use these default names, or you can create your own markers. To reduce complexity, use the default marker names, which are also used throughout the documentation. You can also use the style names defined in the documentation to reduce complexity. The following table lists the default custom marker types used to implement online features.

Marker Type	Description
AbbreviationTitle	Specifies abbreviation alternate text for browsers to display for abbreviations such as SS# when a user hovers over the abbreviation in output. Screen readers also can read the abbreviation alternate text. Used in combination with the Abbreviation character format.
AcronymTitle	Specifies acronym alternate text for browsers to display for acronyms such as HTML when a user hovers over the acronym in output. Screen readers can also read the acronym alternate text. Used in combination with the Acronym character format.
Citation	Specifies the source of a quote using a fully qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. Used in combination with the Citation character format.
Context Plugin	Specifies context plug-ins for Eclipse help systems. Other Eclipse plug-ins can use the context plug-in IDs to call the Eclipse help system. For more information, see “Using Markers to Specify Context Plug-ins in Eclipse Help”.
DropDownEnd	Marks the end of an expand/collapse section. Used in conjunction with an Expand/Collapse paragraph format.
Filename	Specifies the name of an output file for a page or an image.
GraphicScale	Specifies a percentage to use to resize an image, such as 50 or 75 percent, in generated output.
GraphicStyle	Specifies the name of a graphic style defined in a project to apply to an image. This marker type is an internal marker type that is not displayed in Stationery Designer. You cannot create a marker type with a different name and assign it this functionality.
Hypertext	Specifies a link using the <code>newlink</code> and <code>gotolink</code> commands in Adobe FrameMaker. This marker type is a default Adobe FrameMaker marker type ePublisher automatically maps.
ImageAltText	Specifies alternate text for an image. This text is added to the <code>alt</code> attribute of the <code>img</code> tag in the output. Screen readers use this text when you create accessible content.
ImageAreaAltText	Specifies alternate text for clickable regions in an image map. This text is added to the <code>alt</code> attribute of the <code>img</code> tag in the output. Screen readers use this text when you create accessible content.

Marker Type	Description
ImageLongDescByRef	Specifies the path to the file that contains the long description for an image. This text is added to the <code>longdesc</code> attribute of the <code>img</code> tag in the output. Screen readers read this description when you create accessible content.
ImageLongDescNotReq	Specifies that a long description is not required for an image, which bypasses this accessibility check for the image when you create accessible content.
ImageLongDescText	Specifies the long description for an image. This text is added to the <code>longdesc</code> attribute of the <code>img</code> tag in the output. Screen readers read this description when you create accessible content.
Keywords	Specifies the keywords to include in the <code>meta</code> tag for the topic. The <code>meta</code> tag improves searchability on the Web.
PageStyle	Specifies the name of a page style defined in the project to apply to a topic. This marker type is an internal marker type that is not displayed in Stationery Designer. You cannot create a marker type with a different name and assign it this functionality.
PassThrough	Specifies that ePublisher place the contents of the marker directly into the generated output without processing the content in any way. For example, you could use a <code>PassThrough</code> marker if you wanted to embed HTML code within your generated output.
Popup	Specifies the start of the content to include in a popup window. The content is displayed in a popup window when you hover over the link. When you click the link in some output formats, the topic where the popup text is stored, such as the glossary, is displayed.
PopupEnd	Marks the end of the content to include in a popup window.
PopupOnly	Specifies the start of the content to include in only a popup window. Browsers display the content in a popup window when you hover over or click the link.
RubiComposite	No longer supported.
SeeAlsoKeyword	Specifies an internal identifier for a topic. <code>SeeAlsoLink</code> markers in other topics can list this identifier to create a link to this topic. Used in conjunction with a <code>See Also</code> paragraph format or character format.
SeeAlsoLink	Identifies an internal identifier from another topic to include in the list of <code>See Also</code> links in this topic. Used in conjunction with a <code>See Also</code> paragraph format or character format.
SeeAlsoLinkDisplayType	Specifies whether to display the target topics on a popup menu or in a window. By default, the links are displayed in the Topics Found window. To display a popup menu, set the value to <code>menu</code> . This marker type is supported only in HTML Help.
SeeAlsoLinkWindowType	Specifies the name of the window defined in the <code>.hhp</code> file, such as <code>TriPane</code> or <code>Main</code> , that the topic opens in when the user clicks the link. This marker type is supported only in HTML Help.
TableStyle	Specifies the name of a table style defined in the project to apply to a table in versions of Microsoft Word that did not support table styles. This marker type is an internal marker type that is not displayed in Stationery Designer. This marker type is supported only for Microsoft Word documents. You cannot create a marker type with a different name and assign it this functionality.
TableSummary	Specifies an alternate text summary for a table, which is used when you create accessible content. This text is added to the <code>summary</code> attribute of the <code>table</code> tag in the output. Screen readers read this description when you create accessible content.
TableSummaryNotReq	Specifies that a summary is not required for a table, which bypasses this accessibility check for that table.

Marker Type	Description
TOCIconHTMLHelp	Identifies the image to use as the table of contents icon for a topic in the HTML Help output format.
TOCIconJavaHelp	Identifies the image to use as the table of contents icon for a topic in the Sun JavaHelp output format.
TOCIconOracleHelp	Identifies the image to use as the table of contents icon for a topic in the Oracle Help output format.
TOCIconWWHelp	Identifies the image to use as the table of contents icon for a topic in the WebWorks Help output format.
TopicAlias	Specifies an internal identifier for a topic that can be used to create a context-sensitive link to that topic.
TopicDescription	Specifies a topic description for a context-sensitive help topic in Eclipse help systems. For more information, see “Using Markers to Specify Topic Descriptions for Context-Sensitive Help Topics in Eclipse Help”.
WhatIsThisID	Identifies a What Is This help internal identifier for creating context-sensitive What Is This field-level help for Microsoft HTML Help.
WikiCategory	Specifies the category or label you want to assign to a topic when generating Wiki output. For more information, see “Defining Wiki Categories or Labels”.
WindowType	Specifies the name of the window defined in the help project that the topic should be displayed in. In Microsoft HTML Help, the window names are defined in the .hhp file. This marker type is supported in Microsoft HTML Help and Oracle Help.

AutoText, AutoCorrect, and User-Defined Hotkeys

Autotext improves consistency and efficiency for your writing team. Writers can quickly create items, such as tables and lists, with the correct styles applied by default. You can create autotext for many common items:

- 2-, 3-, 4-, and 5-column tables
- Checklists
- Notes, tips, cautions, and warnings
- Cross reference introductory phrases, such as `For more information, see`
- New chapters and standard topics, such as tasks or command reference topics

AutoCorrect allows you to automatically fix common typing mistakes, including two capital letters in a row and misspelled words. You can also use autocorrect to define some shorthand character sequences that automatically insert longer common phrases. For example, you can define `acl` to be replaced with `access control list`. When you type `acl` and a space, Microsoft Word changes it to `access control list`.

Microsoft Word also allows you to define hot keys for common tasks, such as applying a style, inserting autotext, or running a macro. With hotkeys, autotext, and autocorrect, you can create content more efficiently.

Toolbars and Menus in Word

Create custom toolbars with the options you use most often. You can copy toolbar buttons from the Transit menu and other toolbars to create one combined toolbar with drop-down menus and selections based on your specific requirements. Helpful commands to include on your toolbars are Default Paragraph Font, Keep with Next, and Show/Hide field codes.

Variables and Conditions in Word

On the Properties tab, you can define custom variables to use throughout your source documents. To insert a variable in your content, create field code that references the custom property you created. You can also use field codes to display the contents of a specific style type, such as the previous Heading 1.

For conditions in your generated output, you can use the field codes (markers) supported by ePublisher. You can also use paragraph and character styles to identify conditional content. With this style approach, you need to create extra styles for each condition you need. Then, you can create multiple templates that show or hide specific styles. By attaching the appropriate template, you can include or exclude the appropriate content in your printed output. You can also include or exclude content from your generated output based on styles.

Page Layouts and Sections in Word

You should define all the sections you need in your template. Each section is separated by a section break and has its own page setup. The table of contents and index often have multiple section breaks to customize the page layout in those sections and correctly generate the lists based on field codes. You need to be careful when working around section breaks. If you delete a section break, the page layout for the section, including the headers and footers, may be changed.

In the headers and footers, use field codes to display the contents of a specific style from the associated section, such as the Title style from the title page to include the book title in the footer. This type of field code is automatically maintained and updated. If you use a variable field code in the headers or footers, you need to manually update those field codes. Variable field codes in headers and footers are not updated automatically with the rest of the document content.

Table of Contents and Index in Word

Since the appearance of online table of contents and indexes often differ from printed versions, you need to be able to deliver customized table of contents and indexes in your online content. Therefore, ePublisher does not need the table of contents and index formatting defined in your source documents. ePublisher allows you to define the table of contents levels and appearance, as well as the appearance of the index in your generated output. ePublisher uses the index field codes throughout your source documents to build the online index. This support allows you to deliver the online content you require.

Automation with Macros in Word

Macros allow you to automate, customize, and extend the default Microsoft Word capabilities, including some common tasks and maintenance operations:

- Deleting hidden table of contents bookmarks that build up over time
- Production book tasks, such as updating fields and paginating multiple files
- Attaching different templates for conditional text implementations

This automation can streamline the content authoring process and save you valuable time and effort. Microsoft Word provides a VBA (Visual Basic for Applications) environment for macros to give you the automation capabilities you need. To enable macros, set the security level to medium. You can record steps to perform a specific task, and then copy and edit the generated code to do exactly what you need.

Designing DITA Usage Standards

DITA (Darwin Information Typing Architecture) is an XML-based format for creating and publishing technical content. DITA leverages the strengths of XML and provides a standard set of element definitions used to create technical content. Within the topic types based on the standard topic definition, DITA specifies the information elements used to define the content, such as the title, paragraph, table, and list elements.

This section describes the design usage considerations for DITA source documents. By reviewing how ePublisher processes DITA source documents, you can streamline single-sourcing processes and reduce your production and maintenance costs. This section does not describe all DITA details, but it focuses on the design and usage considerations related to ePublisher.

Using DITA Standards to Support Single-Sourcing

ePublisher accepts ditamaps as input source documents. The ditamap identifies the structure and order of the XML files and the DTD. The DTD defines the classes for each element, such as topic/topic and task/task. ePublisher uses these classes to process the content. ePublisher also allows you to add individual XML files as source documents to a project. However, if you publish individual XML files without using a ditamap, links will not be generated in the output.

Note: Publishing individual XML files is only supported when using the DITA-OT version 1.8 and above.

Mapping DITA Classes to ePublisher Styles

The `default.wwconfig` file maps classes to styles in ePublisher, defining both the style type and the style name. You can override the `default.wwconfig` file to specify your own style names and types. The override file does not need to be comprehensive. Your override file can build on the default file. When the same match is defined in both the customized override file and the default file, the match in the customized file overrides the match in default file.

You can override the `default.wwconfig` file for all output formats, a specific output format, or a specific target. The `default.wwconfig` file can also emit WIF information, such as `ww` content for bulleted and numbered list items. For more information, see the default `default.wwconfig` file in the following folder:

```
Program Files\WebWorks\ePublisher Designer\Adapters\xml\scripts\dita
```

For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.

To specify your own class mappings with an override file

1. *If you want to override the class mappings for all output formats*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Format Override Directory**.
 - b. Create the `Formats\Adapters\xml\scripts\dita` folder in your project folder.
2. *If you want to override the class mappings for one output format*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Format Override Directory**.
 - b. Create the `Formats\Adapters\formattype\xml\scripts\dita` folder in your project folder, where *formattype* is the name of the output format you want to override, such as `WebWorks Help 5.0`.
3. *If you want to override the class mappings for a target*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Target Override Directory**.
 - b. Create the `Targets\targetname\Adapters\xml\scripts\dita` folder in your project folder, where *targetname* is the name of the target you want to override.
4. Copy the `default.wwconfig` file from the following folder to the override folder you created within your project folder:

```
Program Files\WebWorks\ePublisher Designer\Adapters\xml\scripts\dita
```
5. Open the `default.wwconfig` file you copied to your project override folder.
6. Remove any mappings you do not want to override.
7. Add the mappings you want to override, including a match statement that defines the style name and a match statement that defines the style type.
8. Save and close the `default.wwconfig` file you copied to your project override folder.

The `default.wwconfig` file provides two sections:

<styles> section

Provides the style match statements that map DITA classes to style names in ePublisher. Each style match statement, such as `<Style match=...`, defines the classes and parent classes to use for a match. When a

class matches the style match statement, the XSL specifies the ePubublisher style name to associate with that class.

<Types> section

Provides the type match statements that map DITA classes to style types in ePubublisher. Each type match statement, such as `<Type match=...`, defines the classes to use for a match. When a class matches the type match statement, the value attribute specifies the ePubublisher style type for the style associated with that class.

Defining Online Features with DITA

You assign formatting and features to styles in ePublisher. These style definitions are stored in your Stationery and mapped to DITA classes. For some online features, such as index entries, you use the standard DITA elements to implement the feature. For more information about creating Stationery and implementing online features, see “Designing, Deploying, and Managing Stationery”.

Review the following considerations to understand how to implement each online feature using DITA elements and the Stationery options within ePublisher:

- For index entries, use the standard DITA tag.
- For related topics, assign related topics options in ePublisher to paragraph styles for related-links elements, such as the Related Task and Related Concept styles.
- For expand/collapse sections, assign dropdown options in ePublisher to the appropriate paragraph styles. You cannot use a marker to identify the end of the expand/collapse section, so you need to use a paragraph style to identify the end.
- For conditions, use attributes and ditaval file filtering, for more information, See “Using Ditaval files in DITA”.
- For variables, use conref and ditaval file filtering.
- For popup windows, use the xref element for the link and define paragraph styles with the popup options in ePublisher.
- For specifying file names, use the othermeta element to define the marker with the name for the file.
- For specifying a topic alias for context-sensitive help links, use the othermeta element to define the marker with the value of the topic ID for that topic.
- For meta tag keywords, use the othermeta element to define the marker with the keywords you want to include in the meta tag in the generated output.
- For opening a topic in a custom window, use the othermeta element to define the marker with the name of the window in which to open the topic.
- For a custom TOC icon, use the othermeta element to define the marker with the name of the TOC icon to use for the generated topic.
- For See Also links, use related topics, which are available in more output formats. See Also link support is being reviewed for future enhancements.
- For accessibility features, such as image alternate text and long descriptions, use standard DITA elements and attributes.

Customizing the DITA DTD

You can associate an instance of the DITA-OT with your Stationery design project to apply and track DITA specialization through the Stationery.

To customize the DITA DTD and apply the DITA specialization

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. Create the `Formats\Helpers\dita-ot-1.4` folder in your *projectname* folder, where *projectname* is the name of your Stationery design project.
3. Copy the contents of the following folder to the override folder you created within your project folder:
`Program Files\WebWorks\ePublisher Designer\Helpers\dita-ot-1.4`
4. Open the `dita-ot-1.4` folder you copied to your project override folder and apply the DTD changes you need for your DITA specialization. Typically this means running the `ant integrate` task to install your DITA-OT plug-ins.
5. Save and close the modified files.
6. Regenerate your project to review the changes.

DITA Specialization

Structure is mapped to style in dita, for more information, please refer to the following website:

[http://wiki.webworks.com/HelpCenter/Tips/DITA/All Versions/DITA Configuration](http://wiki.webworks.com/HelpCenter/Tips/DITA/All%20Versions/DITA%20Configuration)

5

Designing, Deploying, and Managing Stationery

This section outlines the Stationery design, deployment, and management process. ePublisher Express projects use Stationery designed in ePublisher Designer to define the appearance and behavior of generated output. You can use Stationery to define a standard for one or more projects to use. You can also customize the design to meet your specific needs.

Writers use the Stationery when they create projects. When the Stationery designer changes or updates the Stationery a project uses, ePublisher prompts writers to synchronize their projects with the updated Stationery associated with their projects. This powerful feature allows you to quickly deploy and manage Stationery updates across your organization.

Checklist: Design, Deploy, and Manage Stationery

The following checklist outlines the tasks involved in designing, deploying, and managing your Stationery. Many tasks refer you to a section that provides more information about that task. Use this checklist as a guide to develop and manage your Stationery.

Prerequisite Tasks

<input checked="" type="checkbox"/>	Task
<input type="checkbox"/>	1. Review the Stationery-related concepts, such as Stationery design projects and Stationery synchronization. For more information, see “Understanding Stationery”.
<input type="checkbox"/>	2. Create sample source files and templates with all styles and formats, cross-reference formats, variables, conditions, markers, and field codes to be used in your source files. These sample files should have all styles in all possible combinations, and the paragraphs should have text that wraps to allow you to review all format settings and the appearance of each type of paragraph. For more information, see “Designing Input Format Standards”.

Stationery Design Tasks

<input checked="" type="checkbox"/>	Task
<input type="checkbox"/>	1. Create a Stationery design project based on a default output format and add your sample source files to the project. For more information, see “Creating a Stationery Design Project”.
<input type="checkbox"/>	2. Add the other output formats and targets you want to support with your Stationery to your Stationery design project. For more information, see “Adding Output Formats to Your Stationery Design Project”.
<input type="checkbox"/>	3. Scan your sample source files in your Stationery design project to make sure your project has all the elements from your standard source files. For more information, see “Updating a Project to Include All Styles”.
<input type="checkbox"/>	4. Review how Style Designer works and how styles inherit properties so you can save time while designing and maintaining your Stationery. For more information, see “Understanding Style Designer”.
<input type="checkbox"/>	5. Create a hierarchy of styles to allow styles to inherit properties from other styles. For more information, see “Organizing and Managing Styles”.
<input type="checkbox"/>	6. Define how to split content into multiple topics based on styles, such as headings. For more information, see “Defining New Pages (Page Breaks)”.
<input type="checkbox"/>	7. Define the table of contents levels and structure. Also define whether to include links, known as a mini-TOC, at the start of a topic that shows the remaining topics within that subsection. For more information, see “Defining TOCs and Mini-TOCs”.
<input type="checkbox"/>	8. Define the appearance of each paragraph style, such as the font family, font size, and margins. Paragraphs also include notes, tips, warnings, and cautions. For more information, see “Modifying the Appearance of Paragraphs”.
<input type="checkbox"/>	9. Define the appearance of your bulleted and numbered lists. For more information, see “Defining the Appearance of Bulleted Lists”, “Defining the Appearance of Numbered Lists” and “Defining the Appearance of Bulleted and Numbered Lists for Wiki Markup”.
<input type="checkbox"/>	10. Define the appearance of each character style. For more information, see “Modifying the Appearance of Characters”.
<input type="checkbox"/>	11. Define the appearance of your tables. For more information, see “Defining the Appearance of Tables”.
<input type="checkbox"/>	12. Define the appearance of your images, including the format to use for images. For more information, see “Defining the Appearance of Images”.
<input type="checkbox"/>	13. Define the appearance of your pages, such as breadcrumbs and company information. For more information, see “Defining the Appearance of Pages”.
<input type="checkbox"/>	14. Define the color and appearance of links. For more information, see “Defining the Appearance of Links”.
<input type="checkbox"/>	15. Save a backup copy of your Stationery design project. For more information, see “Saving a Snapshot (Backup Copy) of Your Project”.

Optional Online Feature and Customization Tasks

<input checked="" type="checkbox"/>	Task
<input type="checkbox"/>	1. Define your marker types and field codes. You can use the default names, or you can define custom marker types and field codes. For more information, see “Defining the Processing of Markers and Field Codes”.
<input type="checkbox"/>	2. Define how to handle file naming for generated topics (pages) and images. For more information, see “Defining File Names”.
<input type="checkbox"/>	3. Define how to handle context-sensitive help links. For more information, see “Defining Context-Sensitive Help Links”.
<input type="checkbox"/>	4. Define support for expand/collapse sections. For more information, see “Defining Expand/Collapse Sections (Drop-Down Hotspots)”.
<input type="checkbox"/>	5. Define how to handle and support popup windows. For more information, see “Defining Popup Windows”.
<input type="checkbox"/>	6. Define how to handle and support related topics. For more information, see “Defining Related Topics” and “Defining See Also Links”.
<input type="checkbox"/>	7. For each target, define the default target settings and format-specific features. For more information, see “Define the Default Settings for Each Target”.
<input type="checkbox"/>	8. Save and test your Stationery. For more information, see “Saving and Testing Stationery”.
<input type="checkbox"/>	9. Customize your output as needed. For more information, see “Customizing Your Design”.

Stationery Testing and Deployment Tasks

<input checked="" type="checkbox"/>	Task
<input type="checkbox"/>	1. Save and test your Stationery. For more information, see “Saving and Testing Stationery”.
<input type="checkbox"/>	2. Resolve any remaining issues and customization requirements.
<input type="checkbox"/>	3. Deploy your Stationery and standard source documents. For more information, see “Deploying Stationery”.
<input type="checkbox"/>	4. Maintain your Stationery and deploy adjustments as needed. For more information, see “Managing and Updating Stationery”.

Understanding Stationery

ePublisher projects use Stationery designed in ePublisher Designer by a Stationery designer to define the appearance and behavior of generated output. **Stationery** is a complete set of processing rules and styles that define all aspects of the output. Writers use the Stationery created by the Stationery designer when they create projects and generate output. Once the Stationery designer changes or updates the Stationery, ePublisher prompts writers when they open projects associated with that Stationery to synchronize their projects with the updated Stationery.

Stationery design projects are not based on Stationery. These projects are created in ePublisher Designer and are used to create and maintain Stationery. To modify or update Stationery, you need to update the Stationery design project and then save it as Stationery.

Stationery stores all the style and behavior settings. Stationery also captures the transformation process and isolates it from changes in future ePublisher releases. Since you can always use ePublisher Designer to create new Stationery, you can easily maintain the existing Stationery and move forward with a new ePublisher release. You can also limit the number of formats included in a Stationery to reduce complexity and potential confusion in your working environment. Each output format, such as HTML Help, WebWorks Help, and Dynamic HTML, requires certain files to generate output. Without these files, the formats do not have the components required to generate the correct output.

To create a new project, ePublisher Express users must have Stationery. For more information about Stationery and how it works, see “Understanding Stationery, Projects, and Overrides”.

Stationery Components

Stationery includes the following components:

Stationery .wxsp file

Contains the style definitions and target settings for the selected output formats and targets. This file stores the style definitions including paragraph, character, table, page, graphic, and marker styles. This file also stores the conditions, variable values, cross-reference definitions, and target settings for each target.

Manifest file

Identifies the files in the `Files`, `Formats`, and `Targets` folders associated with the Stationery. Any time a Stationery designer adds, removes, or modifies a file in one of these folders and saves the Stationery, ePublisher updates the Stationery manifest file.

Files folder

Contains all the files and folders you want to include in the project. ePublisher copies these files and folders to the output location and includes them in the output.

Formats folder

Contains a complete copy of all the run-time files required to transform your source documents for specific output formats. In your Stationery design project, this folder contains the customized files for specific output formats needed to transform your source documents and create your output files. This folder includes a subfolder for each defined output format with customized files. These files override the default files used by ePublisher to generate the associated output format. New ePublisher releases may include changes to the default transformation files. By storing the customized files in a separate location, ePublisher can override the default files by using these customized files and it can also protect these customized files when you install a new ePublisher release.

Targets folder

Contains the customized files for specific targets needed to transform your source documents and create your output files. This folder includes a subfolder for each defined target with customized files. These customized files override the files stored in the `Formats` folder for the output format related to the target. For example, you can create a customized `Page.asp` page layout file for a specific target.

Understanding Stationery Synchronization

ePublisher projects use Stationery designed in ePublisher Designer to define the appearance and behavior of generated output. **Stationery** is a complete set of processing rules and styles that define all aspects of the output. Writers use the Stationery created by the Stationery designer when they create projects and generate output.

Once the Stationery designer changes or updates the Stationery, ePublisher prompts writers when they open projects associated with that Stationery to synchronize their projects with the updated Stationery. ePublisher Express prompts you to synchronize your project with its Stationery under the following conditions:

- The project manifest file differs from the Stationery manifest file.
- The Stationery settings have been modified.

When writers synchronize their projects with the updated Stationery, all settings in the project are updated to match the Stationery. For more information, see “Synchronizing Projects with Stationery”.

Designing Stationery

ePublisher projects use Stationery designed in ePublisher Designer to define the appearance and behavior of generated output. You can use Stationery to define a standard for one or more projects to use. You can also customize the design to meet your specific needs. Writers then use the Stationery when they create projects.

There are many considerations when designing and developing Stationery. A well-designed Stationery is less costly to maintain and manage into the future. For more information about the complete design, deployment, and management process, see “Checklist: Design, Deploy, and Manage Stationery”.

Creating a Stationery Design Project

Stationery design projects are not based on Stationery. You create a Stationery design project in ePublisher Designer and use it to create and maintain Stationery. To create Stationery, you need to save a project as Stationery. To modify or update Stationery, you need to update the Stationery design project and then save it as Stationery.

The ePublisher workflow improves productivity and saves time for writers who need to manage online content. You create Stationery that defines a set of reusable settings to apply to any project. Without any additional work, writers can load these settings into their projects, and update them automatically when a change to the Stationery is made.

Note: To create a new Stationery design project from an existing Stationery design project, you can manually copy the existing Stationery design project .wep file and associated files and folders.

To create a Stationery design project

1. Open ePublisher Designer.
2. On the **File** menu, click **New Project**.
3. In the **Project name** field, type the name for your Stationery project.
4. In the **Format** field, select the default output format you want to include in your Stationery, and then click **Next**. You can add other output formats at a later time.
5. Click **Add** and select your standard sample source files to add them to the **Source Documents** list box. These files provide all the standards for each input format.
6. When all your sample source files are listed in the **Source Documents** list box, click **Finish**.
7. On the **Project** menu, click **Scan All Documents** to add all elements defined in your sample source documents to the project. This process ensures you can define how ePublisher processes each of these elements, such as styles and variables.
8. On the **File** menu, click **Save** to save your Stationery design project.

Next, you can add all the other output formats you want to include in your Stationery. For more information, see “Checklist: Design, Deploy, and Manage Stationery”.

Adding Output Formats to Your Stationery Design Project

You can include one or more output formats in your Stationery. To simplify your Stationery, define only the output formats you need. Writers can then use these output formats defined in the Stationery to create one or more targets in their projects.

Each output format creates a target in your Stationery design project. You can also define multiple targets with the same output format. For example, you can define multiple targets, one for each standard OEM partner you may have. Each of these targets may generate the same output format with settings, such as company name and address, customized for each partner.

The Stationery Designer properties and options are shared across all targets and output formats. Some settings, such as the target settings, variable values, conditions, and cross-reference formats, are defined per target. Some targets and output formats also offer additional features and customizations.

Adding a Target to Your Stationery Design Project

When you create your Stationery design project, you select the default output format. ePublisher adds the target for that output format to your project. You can then add more targets and output formats to your project.

To add a target to your Stationery design project

1. Open your Stationery design project.
2. On the **Project** menu, click **Manage Targets**.
3. Click **Add**.
4. In the **Format Type** field, select an output format for the target.
5. In the **Target Name** field, type a name for the target, and then click **OK**.

Selecting an Active Target in Your Stationery Design Project

The **active target** is the target currently selected for your project. ePublisher uses the active target to identify what output to generate. If you modify settings, such as variable values, that are applied to an individual target and are not shared across targets in a project, ePublisher saves those changes for the active target. ePublisher applies the settings defined through the options on the **Target** menu to the active target.

To select the active target

1. Open your Stationery design project.
2. On the **Project** menu, click the target you want in the **Active Target** menu option.

Updating a Project to Include All Styles

ePublisher transforms your source documents into content that you can deliver to virtually any online format or platform. To make sure your content is properly displayed and easy to use for your audience, you may want to modify the appearance of the online content, as well as add features your audience wants and needs. However, you do not want to modify the original source documents multiple times to produce the different types of output you need, such as print, online help, and Web site content.

ePublisher enables you to maintain your source documents as you need to for print delivery. You can then use Style Designer to define how you want your content transformed for your other output formats. ePublisher helps you create the appearance you want and the functionality you need for your online output.

Your project defines how to transform your content based on the paragraph, character, table, graphic, page, and marker styles in your source documents. If you add new elements to your source documents, you need to scan your documents to include these new elements, such as new paragraph styles, in your project.

To scan all your source documents

1. Open your Stationery design project.
2. On the **Project** menu, click **Scan All Documents**.

Understanding Style Designer

Style Designer allows you to modify how various styles appear in your generated output. You can define how paragraphs, characters, tables, and images are displayed. You can also modify other aspects of your content, such as page layout and how page breaks are established. Style Designer defines the appearance of your online content, such as the color or font of a paragraph style, the style of a table border, the layout of a page, and the file format of your images in your generated output.

ePublisher uses the styles in your source documents to define how to transform and present your content online. ePublisher intelligently senses the styles in the source documents and presents a list of these styles in Style Designer. You can then define your generated output by specifying properties and options for each style. By using styles in your source documents, you have great control over your output. In Style Designer, the **Properties** tab controls the appearance of the selected style and the **Options** tab controls the behavior of the selected style in your generated output.

For example, if you are using a style called `Heading 1` in your source document, ePublisher lists this style in your project with the other styles used in your source documents. If you select `Heading 1` from the list of styles in Style Designer, you can then define all content that has that style to appear a specific way in your generated output. These modifications do not affect your source documents. ePublisher is essentially a filter that transforms your source documents into the output format you need.

You can also use a custom CSS file to modify the appearance of your content instead of using Style Designer for HTML-based output formats. For more information, see “Using a Custom CSS to Modify the Appearance of Content”.

Modifying Output with CSS Properties and Attributes

Cascading style sheet (CSS) properties give you detailed control over the appearance of your generated output. Style Designer provides an intuitive interface to help you modify your output using CSS properties. In addition to the online help, additional knowledge of CSS properties and attributes can help you achieve the results you want. For more information about CSS principals, see W3 Schools at www.w3schools.com and the World Wide Web Consortium (W3C) at www.w3.org.

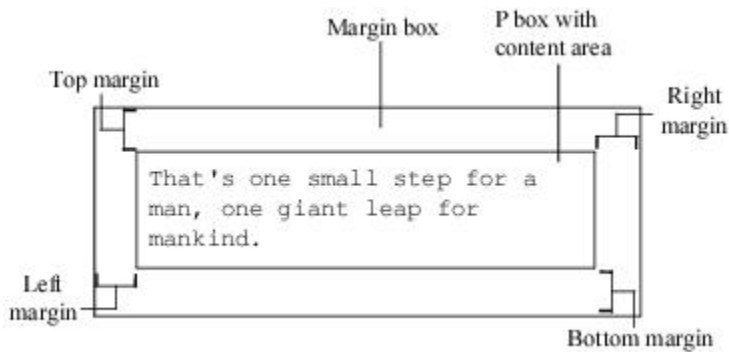
Because your online output is HTML- and CSS-based, the appearance of your output can be inconsistent between different browsers. This inconsistency is a limitation of HTML and CSS, since not all browsers implement the standards defined by the W3C in a consistent way. In addition, some output formats do not support all HTML and CSS features. In these cases, ePublisher disables or hides the incompatible options.

Understanding the CSS Box Model

In accordance with the CSS2 specification, each element on a Web page generates its own invisible, rectangular box, sometimes referred to as the box model. This box delimits the space that the element occupies on the page and consists of a content area surrounded by optional blocks related to the border, margin, or padding around the content. For example, consider the following HTML snippet of a paragraph on a Web page:

```
<p>That's one small step for a man, one giant leap for mankind.</p>
```

Since this content is a paragraph, the text uses a `<p>` tag and is contained in a `P` box, or paragraph box. Assuming this paragraph has only a margin applied to it (and no border or padding), its box can be represented as shown in the following figure:



Other paragraph blocks, heading blocks, table blocks, and so forth may precede or follow this paragraph, and all blocks fit together inside a page block, which contains all other elements of the page. Boxes usually correspond to HTML `<h>`, `<p>`, or `<div>` tags and often contain inline elements within them. Inline formatting, such as a bold word within a sentence, creates inline invisible boxes within the paragraph text.

For Western languages, the boxes, and therefore the content within them, are arranged in a left-to-right, top-to-bottom flow, which is called normal flow. You can reverse normal flow for Eastern languages that flow from right to left. The normal flow includes all styles unless you move them outside of the normal flow using the **Float** or **Positioning** properties. With the **Float** or **Positioning** properties, you can create multi-column pages and other layouts that, in CSS1, required you to use tables. For more information about the CSS2 visual formatting model and normal flow, see the W3C documentation at www.w3.org/TR/REC-CSS2/visuren.html.

Inheriting Style Properties

Many elements of your online design may have similar properties. For example, paragraphs, bulleted list items, and numbered list items may all use the same font and vertical spacing. You may also identify elements that should use settings from your source documents. For example, you may want tables to use the size defined in your source files. Style Designer allows you to specify a precise value for a property, or you can specify from where a property inherits its value. The inheritance options are defined as follows:

Explicit

Ignores all inheritance values and uses the value you specify for this property.

Do not emit

Excludes the property from the generated output for the selected style. This option can help you troubleshoot a design issue and determine which property or element is associated with the issue.

Inherit from style

Inherits the value for the property from the parent style. You can organize styles in a hierarchy and then use inherited properties to reduce maintenance costs for future changes. For example, if you have a Heading 1 and a Heading 2 style, you could make Heading 2 a child of Heading 1. If you select **Inherit from parent style** for a property of Heading 2, it inherits the value for that property from Heading 1. For more information, see “Organizing and Managing Styles”.

Document paragraph style

Inherits the value for the property from the style definition in your source document. By choosing this option for a property, ePublisher uses the formatting from your source document for that property.

Document style catalog

Inherits the value for the property from the style definition in the source documents. When you select this option for a property, ePublisher uses the source document definition and ignores all manual changes made to the style in the source document.

Understanding Options in Style Designer

The **Options** tab allows you to define the behavior of the selected item in Style Designer. For example, you can set the options for a paragraph style to create an expand/collapse section or to split the content and start a new output file. The available options depend on the selected item, such as a paragraph style or a marker style, and the active target, such as WebWorks Help or Eclipse Help.

Note: The options you specify for a parent style are *not* inherited by child styles.

Organizing and Managing Styles

In Style Designer, you can organize your styles in a hierarchy and then use inherited properties to reduce design and maintenance time. You can create a hierarchy of similar styles, set the style property values once, and have those values inherited by child styles.

The **Prototype** style, which is essentially the parent style for all styles, allows you to quickly define properties for all styles. This style allows you to make global changes across all styles that inherit properties from the **Prototype** style. For example, you may have all styles inherit their font and vertical spacing from the **Prototype** style.

Note: Child styles inherit only property values from a parent style. Child styles do *not* inherit additional features specified on the **Options** tab.

By default, all style properties of a parent style are inherited by its child styles. If you change the property values of a parent style, those changes are inherited by the child styles of that parent style. You can override specific properties for a child style to make those not inherited from its parent style. Once the value of a specific property is set at the child level, changes you make to the parent style for that property do not affect the child style.

To organize styles in Style Designer

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. Click the type of styles you want to organize, such as **Paragraph Styles**.
4. *If you want to make a style the child of another style*, drag and drop the child style on top of the style you want to make its parent style.
5. Click on the child style and review the properties for that style to make sure it inherits the property values you want from its parent style.

Previewing the Output from a Source File

Once you specify or change settings in your Stationery design project, you should generate your output to review and verify your changes. You can also generate output for an individual source document. As you develop your Stationery, you will often modify a few settings, generate your output and review the results, and then continue the process until you have the final results you want.

To quickly review your changes, you can also display a preview of a source document. A **preview** allows you to quickly see the affects of the changes you made, without requiring you to generate output for the project. The preview provides a close approximation of how your generated output will look, but your generated output may not exactly match the preview.

Note: At intervals while you develop your Stationery, you may want to create a backup copy of your Stationery design project to serve as a snapshot. This backup gives you a version to return to and use if you make some changes that you no longer want. For more information, see “Saving a Snapshot (Backup Copy) of Your Project”.

To display a preview of a source document

1. Open your Stationery design project.
2. In **Document Manager**, select the source document for which you want to generate a preview.
3. On the **Project** menu, click **Display Preview**.

ePublisher displays a preview of the source document on a preview tab labeled with the name of the source document you selected. The preview provides you with an idea of how modifications you made to project settings affect the appearance of your output. However, some output features are not displayed or active within the preview, such as popup windows, links, and conditions.

Defining New Pages (Page Breaks)

By default, ePublisher transforms each source document into one output page. You can associate a page break with specific paragraph styles to split your content into multiple pages, where each page creates a new output file. By dividing your content, you can present information to your audience in smaller chunks, organize and focus your content, reduce redundant information through links, and reduce scrolling by your audience. To avoid empty topics when multiple heading styles occur in a row, you can also define page break handling based on whether the previous style created a new page. This flexibility enables you to deliver your content your way.

To create new pages based on styles

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Paragraph Styles**, select the style for which you want to create a page break.
5. On the **Options** tab, assign a value to the **Page break priority** option. For more information about this option, click **Help**.
6. Review the target settings to ensure the **Page break handling** setting will correctly process your priority level values by completing the following steps:
 - a. On the **Project** menu, select the **Active Target** you want to specify settings for.
 - b. On the **Target** menu, click **Target Settings**.
 - c. Set the **Page break handling** setting to the appropriate value. For more information about this setting, click **Help**.

Defining TOCs and Mini-TOCs

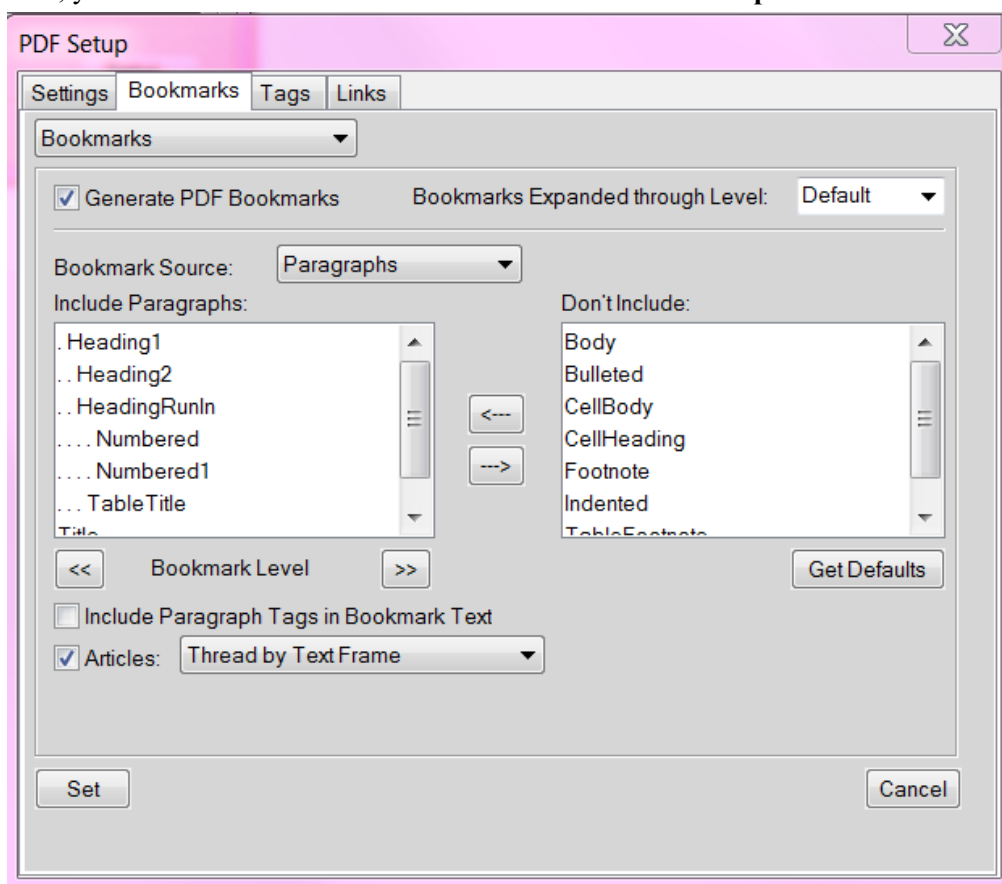
ePublisher does not use the table of contents (TOC) in your source document when it generates the online table of contents. This process allows ePublisher to correctly apply conditions and variables to the TOC, and to format the TOC as needed for the output format. ePublisher bases the TOC on paragraphs from your source document. Depending on the output format you select, you can specify whether to generate the TOC and what file name to assign to the TOC. For more information, see “Generating and Naming the Table of Contents File”. You can also modify the appearance of the table of contents for some output formats. For more information, see “Customizing the Navigation Pane in WebWorks Help” and “Modifying the Appearance of the Table of Contents in Dynamic HTML”.

Defining the Table of Contents Structure (Levels)

By default, ePublisher attempts to automatically determine your source content's TOC levels. You can manually adjust those levels if the settings do not meet your requirements.

To define the table of contents structure (levels)

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Paragraph Styles**, select the style you want to include in your TOC.
5. On the **Options** tab, specify a value for the **Table of contents level** option. The default value, **Auto-Detect**, attempts to infer the necessary information based on your source content. Specify a value that indicates the level of the table of contents entry for the selected paragraph style. When working with Adobe FrameMaker, you can determine the levels by examining the **PDF Bookmark** levels for paragraphs that appear in the table of contents. Here is an example of the **Bookmarks** tab from FrameMaker 11, to navigate here, you would select **File > Print....** Then click the **PDF Setup...** button.

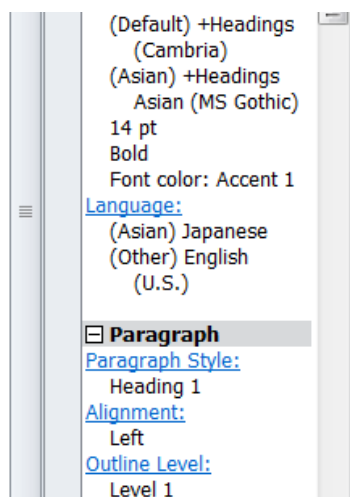


When working with Microsoft Word sources, the auto-detect values are based on the **Outline level** of each paragraph style. Below is a screenshot of the Word 2010 interface for each paragraph's outline level, to navigate here, you use the key combination **Shift + F1**.

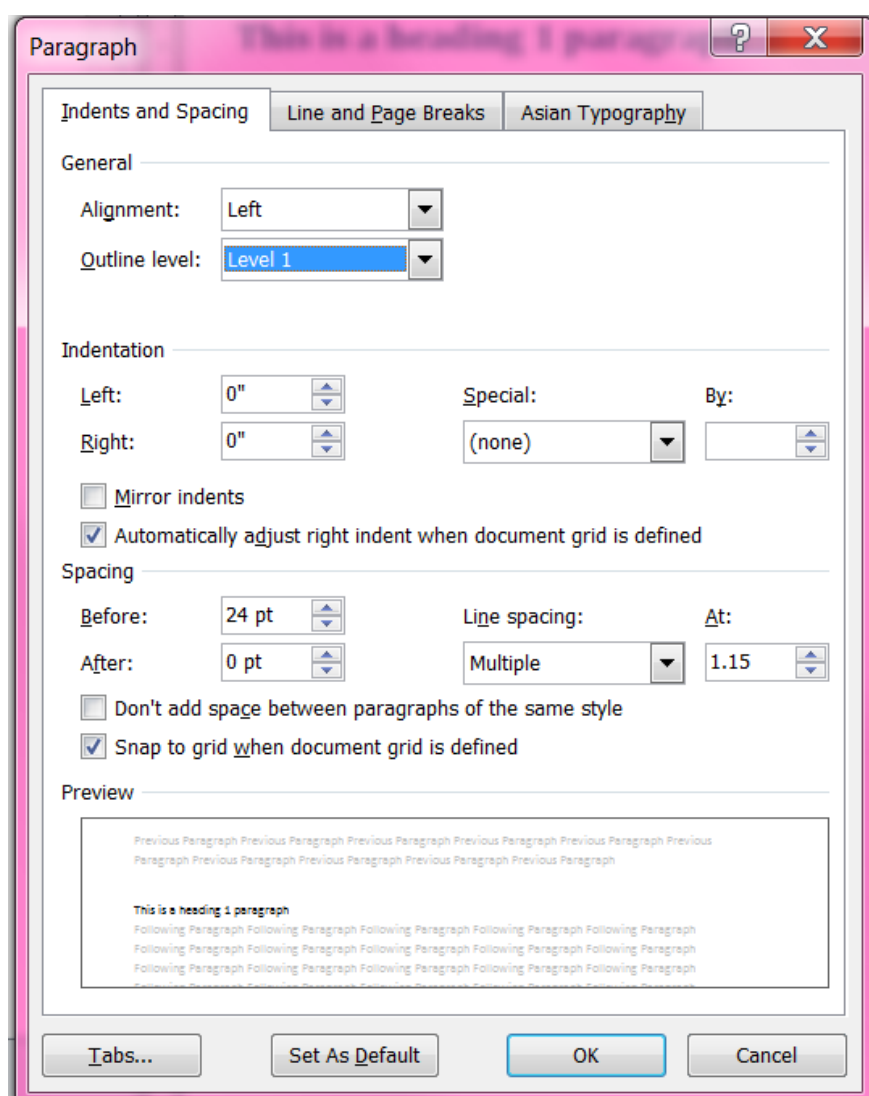
This is a heading 1 paragraph

This is a heading 2 paragraph

This is a heading 3 paragraph



Once you click the **Outline Level** hyperlink, you can select the desired level from the following window.



DITA XML auto-toc is taken from the map hierarchy. A good example would be from the sample input provided in the *Exp_Design* project located in the ePublisher directory in the Documents library, or by the [program files]\WebWorks\ePublisher\[version]\Helpers\dita-ot\samples

Generating and Naming the Table of Contents File

By default, once you have defined your table of contents structure using Style Designer, ePublisher generates the table of contents for your output. The Dynamic HTML and Simple HTML formats provide additional capabilities for users to prevent generation of the table of contents or to change the file name of the generated table of contents.

In the **File Processing** section of the target settings, you can also specify whether to include the table of contents, front matter, and index from your source documents in your generated output.

To specify additional table of contents target settings

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. *If you want to prevent ePublisher from generating the table of contents*, set **Generate table of contents** to **Disabled**.
4. *If you want to change the file name of the generated table of contents*, specify the file name you want in the **Table of Contents filename** field.

Defining the Table of Contents from an Irregular Heading Hierarchy

Many authors follow a regular heading hierarchy in their source documents. A **regular heading hierarchy** is one in which no levels are skipped in the hierarchy, as shown in the following example:

First Heading 1
First Heading 2
First Heading 3
Second Heading 3
Second Heading 2
Another Heading 1
Another Heading 2
Yet Another Heading 2

Note: If you use a regular heading hierarchy, this section does not affect your output.

Some documentation methodologies and processes require authors to skip levels in the hierarchy, which produces irregular heading hierarchies like the one shown in the following example:

First Heading 1
First Heading 4
Second Heading 4
First Heading 3
Second Heading 1
Another Heading 3
Still Another Heading 3
Final Heading 1

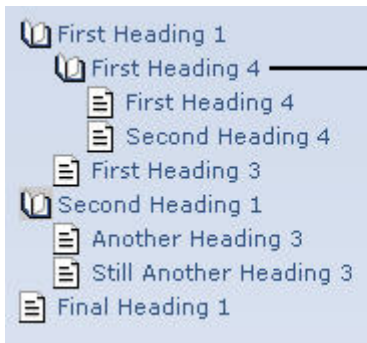
This heading hierarchy in a source document, produces an irregular table of contents hierarchy. The previous example is irregular because all `Heading 2` levels are skipped and several `Heading 3` levels are skipped in the hierarchy.

Since your generated table of contents is created from paragraphs in the source documents, irregular hierarchies can cause an aesthetically displeasing table of contents in your output. ePublisher provides several settings for how to transform irregular heading hierarchies, such as **Fully collapse**, **Don't collapse**, **Smart collapse**, and **Re-label**. By default, ePublisher uses **Smart collapse**.

To specify how ePublisher transforms irregular heading hierarchies

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.

3. *If you want to eliminate skipped levels from the table of contents except where needed to preserve the correct relative hierarchy*, set **Collapse table of contents** to **Smart collapse**. ePublisher removes skipped heading levels from the table of contents. However, if removing skipped levels results in, for example, a Heading 3 and Heading 4 displaying at the same level, ePublisher automatically inserts an entry into the table of contents to preserve the correct relative hierarchy. The irregular hierarchy in the previous example would be generated as follows.



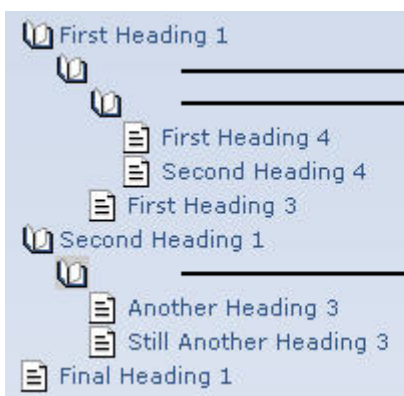
Extra table of contents entry automatically inserted to ensure that Heading 4 is nested deeper than Heading 3.

4. *If you want to eliminate all skipped levels from the table of contents*, set **Collapse table of contents** to **Fully collapse**. ePublisher removes all skipped levels, regardless of the level at which they appear in the source documents. The irregular hierarchy in the previous example would be generated as follows.



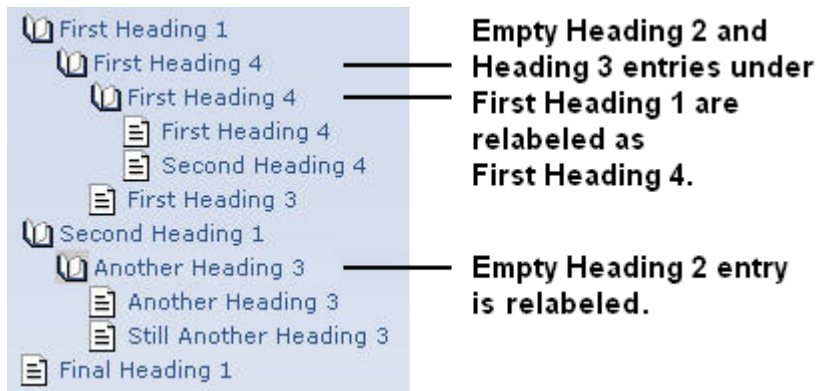
Table of contents is fully collapsed, which allows the Heading 4 and Heading 3 to appear at the same level.

5. *If you want to duplicate the heading hierarchy of the source document with empty entries in the table of contents*, set **Collapse table of contents** to **Don't collapse**. ePublisher automatically inserts empty table of contents entries for the skipped levels. The irregular hierarchy in the previous example would be generated as follows.



Empty table of contents entries are automatically inserted where heading levels have been skipped.

6. *If you want to duplicate the heading hierarchy of the source document with labels in the table of contents*, set **Collapse table of contents** to **Re-label**. ePublisher automatically inserts labeled entries for the skipped levels. The heading text from the entry level appearing beneath the current entry is used as the label. The irregular hierarchy shown in the previous example would be generated as follows.



Understanding the Table of Contents and Merge Settings

Not only can the table of contents be generated from defining your heading paragraph styles, the table of contents can also be generated using Merge Settings. Merge Settings allow you to combine multiple top-level groups into a single hierarchy. Not all output formats support merging. For more information see “Merging Help Systems (Multivolume Help)”.

Note: Since your Stationery design project is used to create Stationery, and Merge Settings are not stored in Stationery, you do not need to configure Merge Settings for your Stationery design project. You can configure the Merge Settings in your ePublisher Express projects, since these settings vary based on individual project needs.

Defining the Icon for a Table of Contents Entry

For some output formats, ePublisher allows you to add a marker to a heading that defines the icon to use for the table of contents entry for that heading. For example, you can use a unique icon for new topics, or for specific types of information. ePublisher provides the TOCIconWWHelp, TOCIconHTMLHelp, TOCIconJavaHelp, and TOCIconOracleHelp markers. For more information about using these markers in source documents, see the *ePublisher Writer Guide*.

Creating a Miniature Table of Contents

A miniature table of contents (mini-TOC), also known as a partial table of contents, provides a list of the topics in the upcoming section. The topic titles are displayed as links beneath the current topic heading for easier navigation within the section. By default, ePublisher does not create a mini-TOC in topics within your output. You can configure ePublisher to generate mini-TOCs and you can define your mini-TOC levels in Style Designer. Before you begin, define the main table of contents for your projects. For more information, see “Defining the Table of Contents Structure (Levels)”. Your mini-TOC is derived from the table of contents level settings you define for the project.

To create a mini-TOC

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Paragraph Styles**, select the style you want to include a mini-TOC after.
5. On the **Options** tab, specify a value for the **Mini-TOC level** option. Specify a value that indicates the levels of the table of contents entries to include in the mini-TOC that follows the selected paragraph style. For more information about this option, click **Help**.

Modifying the Appearance of Mini-TOC Entries

ePublisher stores CSS settings that control the appearance of mini-TOC entries in the `webworks.css` file. You can create an override file to modify these settings for specific levels of the mini TOC or for the entire mini TOC. For example, you can define a different font size and margin for each level in the mini TOC.

For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.

To modify the appearance of the mini TOC

1. *If you want to override the CSS settings for an output format*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Format Override Directory**.
 - b. Create the `Formats\formattype\Pages\css` folder in your project folder, where `formattype` is the name of the output format you want to override, such as `Microsoft HTML Help 1.x`.
2. *If you want to override the CSS settings for a target*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Target Override Directory**.
 - b. Create the `Targets\targetname\Pages\css` folder in your project folder, where `targetname` is the name of the target you want to override.
3. Copy the `webworks.css` file from the following folder to the override folder you created within your project folder:

```
Program Files\WebWorks\ePublisher Designer\Formats\formattype\Pages\css
```
4. Open the `webworks.css` file you copied to your project override folder.
5. *If you want to modify the appearance of the entire mini TOC*, find the code for `div.WebWorks_MiniTOC` and modify the values specified within the braces:
 - To modify the background color, specify the desired RGB color value for the `background-color` option.
 - To modify the border color, specify the desired RGB color value for the `border-color` option.
 - To modify the border width, specify the number of pixels you want for the `border-width` option.
 - To modify the type of border, specify the appropriate border style value for the `border-style` option.
 - To modify the spacing between the border and the text, specify the number of pixels you want for the `padding` option.
 - To modify the font, specify the name of the font you want for the `font-family` option.
6. *If you want to modify the appearance of a specific level of the mini TOC*, find the code for `div.WebWorks_MiniTOC_Levelx`, where `x` is the level number you want to modify. Then, specify the values within the braces to modify the font or margin:
 - To modify the font of all mini-TOC entries for the specified level, specify the name of the font you want, such as `font-family: Arial;`.
 - To modify the font size of all mini-TOC entries for the specified level, specify the size of the font you want, such as `font-size: 14pt;`.

- To modify the left margin indent of all mini-TOC entries for the specified level, specify the indent you want, such as `margin-left: 10px;`.

7. Save the `webworks.css` file.

8. Regenerate your project to review the changes.

For example, the following figure illustrates how you could customize your mini-TOC entries.

```
div.WebWorks_MiniTOC_Level1
```

```
{ font-size: 14pt;
```

```
font-family: Arial;
```

```
margin-left: 6px;
```

```
}
```

```
div.WebWorks_MiniTOC_Level2
```

```
{ font-size: 12pt;
```

```
font-family: Arial;
```

```
margin-left: 16px;
```

```
}
```

```
div.WebWorks_MiniTOC_Level3
```

```
{ font-size: 10pt;
```

```
font-family: Arial;
```

```
margin-left: 16px;
```

```
}
```

Modifying the Appearance of Paragraphs

One of the most common modifications for online output is changing the appearance of text. Using Style Designer, you can select a paragraph style and then define the appearance of that style. This efficient method allows you to globally change all paragraphs with the same style at once, which increases your control and reduces maintenance costs.

The Prototype Style for Paragraphs

The **Prototype** style is the parent to all other styles. When you set a property for the **Prototype** paragraph style, other paragraph styles inherit the value of that property. You can then override that value for specific styles as needed. The **Prototype** paragraph style allows you to quickly change a default property and apply that change to all paragraph styles within your Stationery project.

Depending on how you organized your styles in Style Designer, you may not have one style as the parent style on which all others are based. In Microsoft Word, usually all styles are based on **Normal**, but that may not always be the case. The **Prototype** style ensures that each of your styles has a parent style within your Stationery project.

Setting the Background Color of a Paragraph

In terms of the CSS box model, the background for a paragraph refers to the background of the content and the padding areas. If you increase the padding for a paragraph style, the background color area for that style also increases. If you decrease the padding for a paragraph style, the background color area for that style also decreases.

To set the background color of a paragraph

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Paragraph Styles**, select the style you want to modify.
4. On the **Properties** tab, click **Background**.
5. In the **Color** field, select a color from the drop-down menu, or type the RGB value of the color you want, such as `FFFFFF`.

Setting the Border Style and Color of a Paragraph

Borders are lines that can be drawn around any or all of the four sides of a paragraph. In terms of the CSS box model, increasing the padding for a paragraph style increases the space between the paragraph and the border.

Not all browsers display border styles the same way. For example, some browsers may not differentiate dotted lines from solid lines. The size and spacing of the dots in a dotted line may also be different between various browsers and operating systems.

To set the border style and color of a paragraph

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Paragraph Styles**, select the style you want to modify.
4. On the **Properties** tab, click **Border**.
5. Click the tab for the side of the paragraph you want to display a border, and then specify the color, style, and width for that border. For more information about a property, click **Help**.

Setting the Font for a Paragraph

Setting fonts for online output is an important step in making sure your content is properly displayed for your audience. Because many browsers and help systems use only the fonts available on the user's computer, you may not be able to use specific fonts, such as Times New Roman, as some computers may not have those fonts installed. You can specify a font family, such as sans-serif, to ensure a font of a similar type is used on each computer. You can also specify multiple fonts, separated by commas, to allow the browser to display the first available font.

To set the font of a paragraph

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Paragraph Styles**, select the style you want to modify.
4. On the **Properties** tab, click **Font**.
5. Specify the family, size, style, and other properties you want to modify. For more information about a property, click **Help**.

Setting the Width, Height, and Positioning of a Paragraph

In regards to the CSS box model, modifying the width and height of a paragraph adjusts the content box, which essentially defines the space that the paragraph uses. For example, if an existing paragraph stretches across the entire page, but you then adjust the width to 200 px, the paragraph is then limited to only 200 pixels wide.

To set the width, height, and positioning of a paragraph

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Paragraph Styles**, select the style you want to modify.
4. On the **Properties** tab, click **HTML**.
5. Specify the appropriate values for the width, height, and positioning properties. For more information about a property, click **Help**.

Adjusting the Space Around a Paragraph

You can adjust the white space around a paragraph by adjusting the margin and the padding. In terms of the CSS box model, modifying the padding property adjusts the space inside the border area. For example, if you create a border or background color for a paragraph and you increase the size of the padding, the border moves away from the text in the paragraph.

Modifying the margin properties adjusts the space outside the border area. For example, if you create a border or background color for a paragraph and you increase the size of the margins around the paragraph, the border remains the same distance from the text in the paragraph. However, the position of the paragraph changes since there is more white space between the modified paragraph and the other elements on the page. Review the following additional considerations:

- Set the right margin on paragraph styles to provide spacing between the text and the right edge of the window.
- For paragraphs used within bulleted and numbered lists, carefully adjust the left margin to correctly align with the text of the bulleted and numbered list items. Those list items may have different left margins to leave space for the bullet or number. For more information, see “Defining the Appearance of Bulleted Lists” and “Defining the Appearance of Numbered Lists”.
- When defining the left margin for a paragraph style used in a table, consider the size and weight of the font as well. For example, bold table headings need less pixels in their left margin than non-bold table text paragraphs so that both types of text align with each other.

To set the margin and padding of a paragraph

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Paragraph Styles**, select the style you want to modify.
4. On the **Properties** tab, click **Margin**.
5. Specify a value and select the unit of measure for the **Left**, **Right**, **Top**, and **Bottom** margin properties.
6. On the **Properties** tab, click **Padding**.
7. Specify a value and select the unit of measure for the **Left**, **Right**, **Top**, and **Bottom** padding properties.

Setting the Text Color and Other Characteristics of a Paragraph

In general, ePublisher generates output based on the properties of the content from the original source documents. You can modify the appearance of your online content without modifying your source documents. The text properties allow you to modify several important characteristics of the text:

- Color
- Kerning between letters and words
- Space between lines of a paragraph, also known as letting or line height
- Underlining, overlining, and strikethrough text
- Horizontal and vertical alignment of a paragraph
- Indentation of a paragraph

To set the text characteristics of a paragraph

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Paragraph Styles**, select the style you want to modify.
4. On the **Properties** tab, click **Text**.
5. Specify the appropriate values for the text-related properties, such as **Color** and **Line height**. For more information about a property, click **Help**.

Modifying Paragraphs for Bidirectional Languages

In some documents, such as those written with Arabic or Hebrew script, and in some mixed-language contexts, text within a single paragraph may appear with mixed directionality. For example, some characters are read from left to right, while others within the paragraph may be read from right to left. This phenomenon is called **bidirectionality**, or **bidi** for short.

If a document contains right-to-left characters, and if the browser is able to display the language with the proper character set, the browser must apply the bidirectional algorithm. The proper character set means to not display arbitrary substitutes such as a question mark, a hex code, or a black box for some characters.

ePublisher allows you to make sure the browser correctly displays the content by offering the Unicode **Direction** and **Unicode Bidi** (bidirectional) properties.

To set the Unicode properties of a paragraph

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Paragraph Styles**, select the style you want to modify.
4. On the **Properties** tab, click **Text**.
5. Under **Unicode**, set the appropriate values for the **Direction** and **Unicode Bidi** properties. For more information about these properties, click **Help**.

Disabling Autonumbering in Output

In your source documents, you may have autonumbering enabled for print delivery purposes, such as printed manuals and PDFs. This function allows you to add autogenerated numbers to your chapters, headings, and figure captions. However, for some online delivery, you may not want to include the autonumbering in your online help.

To disable autonumbering

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Paragraph Styles**, select the style that has the autonumbering.
5. On the **Options** tab, set the **Keep paragraph numbering** option to **Disabled**.
6. Repeat steps 4-5 for every paragraph style for which you want to disable autonumbering.

Defining the Appearance of Notes, Tips, Cautions, and Warnings

If you have the word `note`, `tip`, `caution`, or `warning` at the start of the paragraph, and the paragraph is defined with a hanging indent, increase the **Left** margin property in ePublisher to correctly align the word with the appropriate paragraph styles. ePublisher uses the **Left** margin property to define the hanging indent margin.

You may want to add an image to the left of each note, caution, tip, or warning in your generated output. ePublisher does not use images from Adobe FrameMaker reference pages. You can use the **Bullet** properties of a paragraph format to insert an image to the left of the paragraph. This process is similar to using an image for the bullet itself.

Adjusting Paragraph Styles for Wiki Markup

If you plan to generate output for Wiki markup, ePublisher provides the following paragraph style options for Confluence and MoinMoin Wiki output formats:

Wiki Heading Level

Specifies the Wiki heading level of the selected paragraph style. For example, setting this option to **1** specifies that all the paragraphs with the selected style appear in the Wiki at the first level in the Wiki hierarchy. Setting this option to **2** specifies that all paragraphs with the selected style appear in the Wiki at the second level. The value you specify for each of your paragraph styles determines the structure of the Wiki output on the Wiki.

Wiki Indent

Specifies the level of Wiki indent you want to assign to the selected paragraph style. For example, setting this option to **1** specifies that all the paragraphs with the selected style appear in the Wiki at the first indent level in the Wiki hierarchy. Setting this option to **2** specifies that all paragraphs with the selected style appear in the Wiki at the second indent level.

Wiki inline comment

Specifies if the selected paragraph style should be embedded as inline comments on the Wiki. For example, Confluence and MoinMoin Wikis allow users to embed inline comments on Wiki pages. These inline comments can then be displayed or hidden using the Show/Hide Comments control at the top of the Wiki page. If you enable this option for a paragraph style, ePublisher generates the content as inline comments which can then be displayed or hidden on Wiki pages.

To adjust paragraph styles for Confluence and MoinMoin Wiki markup

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Paragraph Styles**, select the style you want to modify.
5. On the **Options** tab, complete the following steps:
 - a. *If you want to specify a heading level for the selected paragraph style*, specify a value for the **Wiki Heading Level** option that indicates the heading level you want the selected paragraph style to have on the Wiki when you deploy your generated output to the Wiki.
 - b. *If you want to specify the level of Wiki indent for the selected paragraph style*, specify a value for the **Wiki Indent** option that indicates the indent level you want the selected paragraph style to have on the Wiki when you deploy your generated output to the Wiki.
 - c. *If you want to specify the selected paragraph style embed content as inline comments on the Wiki*, specify **Enabled** for the **Wiki inline comment** option.

Defining the Appearance of Bulleted Lists

By default, ePublisher generates output for bulleted lists based on the properties of the content from the original source documents. Some tools use the Symbol or Wingdings font for bullets. These fonts may not be available on all computers, which can cause the bullets in your output to not be displayed correctly. You can address this issue by using an image for your bullets, or you can create a character style with a font family assigned, such as sans-serif, and then apply that character style through ePublisher to the characters you choose to define for your bullets.

You may also need to increase the left margin of your bulleted list item styles to provide the correct spacing for the bullet and to allow the text to be properly aligned. In addition, you may need to adjust the left margins of both bulleted and numbered list items to get them to line up with each other. This alignment can reduce the number of vertical columns on a page, which can help the user scan the information. This alignment can also let you use the same paragraph formats for paragraphs within either a bulleted or numbered list. For more information about additional margin adjustments, see “Defining the Appearance of Numbered Lists”.

Note: *If you plan to generate output for Wiki markup*, the process used to define the appearance of bulleted lists for Wiki markup is different than the process used to define the appearance of bulleted list items in other output formats. For more information, see “Defining the Appearance of Bulleted and Numbered Lists for Wiki Markup”.

To create custom bullets

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Paragraph Styles**, select the style you want to modify.
5. On the **Properties** tab, click **Bullet**.
6. *If you want to create a custom bullet based on an image*, complete the following steps:
 - a. Make sure the image you want to use as your bullet image is stored in the **Files** folder in your project. You can copy the image file to this folder.
 - b. In the **Image** field under **Bullet Info**, select the image you want to use.
7. *If you want to create a custom bullet based on a special character*, complete the following steps:
 - a. In the **Text** field, type the special character you want to use. You can type a special character using the Windows Alt key code.
 - b. In the **Separator** field, type the separator characters, such as a non-breaking space, that you would like to use between the bullet and the beginning of the first line of text.
 - c. In the **Character Style** field, select the defined character style you want to apply to the special character specified in the **Text** field. You can specify any character style that already exists in your project. The character style should use a font available on your users’ computers. Avoid custom fonts, which can cause bullet characters to be displayed differently on various computers.

Defining the Appearance of Numbered Lists

When you define the appearance of numbered list items, you need to define margins that leave the correct alignment for the numbers and the hanging indent. Since numbered items have a hanging indent, ePublisher uses a table in the generated output to create the appropriate format. Increase the **Left** margin property in ePublisher to correctly align the the hanging indented text. Then, set the **Flow Indent** text property for the paragraph to a negative value to leave enough space for double-digit numbered steps. For example, if your standard paragraph **Left** margin property is 30 pixels, you may want to set the **Left** margin property for your numbered list items to 70 pixels and set the **Flow Indent** text property to -30 pixels for your numbered list items.

You may also need to work with the left margins of both bulleted and numbered list items to get them to line up with each other. This alignment can reduce the number of vertical columns on a page, which can help the user scan the information. This alignment can also allow you to use the same paragraph formats for paragraphs within either a bulleted or numbered list.

Note: *If you plan to generate output for Wiki markup*, ePublisher provides special options that allow you to specify Wiki list levels and list types for paragraph styles. For more information, see “Defining the Appearance of Bulleted and Numbered Lists for Wiki Markup”.

Defining the Appearance of Bulleted and Numbered Lists for Wiki Markup

If you plan to generate output for Wiki markup, the process used to define the appearance of bulleted and numbered lists for Wiki markup is different than the process used to define the appearance of these items in other output formats. ePublisher provides the following paragraph style options for defining the appearance of bulleted and numbered lists for Confluence and MoinMoin Wiki output formats:

Wiki List Level

Specifies the level of Wiki indent you want to assign to the selected list paragraph style. For example, setting this option to **1** specifies that all lists with the selected paragraph style appear in the Wiki at the first list level in the Wiki hierarchy. Setting this option to **2** specifies that all paragraphs with the selected style appear in the Wiki at the second list level.

Wiki List Type

Specifies if you want to assign an override to the default list type defined for a paragraph style. For example, setting this option to **Bullet** specifies that all lists with the selected paragraph style appear in the Wiki as bulleted list. Setting this option to **Alpha (uppercase)** specifies that all lists with the selected paragraph style appear in the list prefixed with an uppercase letter.

To adjust paragraph styles for Confluence and MoinMoin Wiki markup

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Paragraph Styles**, select the style you want to modify.
5. On the **Options** tab, complete the following steps:
 - a. *If you want to specify a Wiki list level for the selected paragraph style*, specify a value for the **Wiki List Level** option that indicates the list level you want the selected paragraph style to have on the Wiki when you deploy your generated output to the Wiki.
 - b. *If you want to specify a list type for the selected paragraph style*, specify a value for the **Wiki List Type** option that indicates the list type you want the selected paragraph style to have on the Wiki when you deploy your generated output to the Wiki.

Modifying the Appearance of Characters

To modify the appearance of individual characters or words within a paragraph, you can use Style Designer to adjust the appearance of any character styles used in the source documents. This process allows you to optimize styles for print, using the styles in the source documents, and optimize the content for online presentation using the styles defined in ePublisher.

The Prototype Style for Characters

The **Prototype** style is the parent to all other styles. When you set a property for the **Prototype** character style, other character styles inherit the value of that property. You can then override that value for specific styles as needed. The **Prototype** character style allows you to quickly change a default property and apply that change to all character styles within your Stationery project.

Depending on how you organized your styles in Style Designer, you may not have one style as the parent style on which all others are based. The **Prototype** style ensures that each of your styles has a parent style within your Stationery project.

Setting the Background Color of a Character

In terms of the CSS box model, the background for a style refers to the background of the content and the padding areas. If you increase the padding for a style, the background color area for that style also increases.

To set the background color of a character

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Character Styles**, select the character style you want to modify.
4. On the **Properties** tab, click **Background**.
5. In the **Color** field, select a color from the list, or specify the RGB value of a color, such as `FFFFFF`.

Setting the Border Style and Color of Characters

Borders are lines that can be drawn around any or all of the four sides of a style. In terms of the CSS box model, increasing the padding for a style increases the space between the content and the border.

Not all browsers display border styles the same way. For example, some browsers may not differentiate dotted lines from solid lines. The size and spacing of the dots in a dotted line may also be different between various browsers and operating systems.

To set the border style and color of a character

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Character Styles**, select the character style you want to modify.
4. On the **Properties** tab, click **Border**.
5. Click the tab for the side of the character you want to display a border, and then specify the color, style, and width for that border. For more information about a property, click **Help**.

Setting the Font for a Character

Setting fonts for online output is an important step in making sure your content is properly displayed for your audience. Because many browsers and help systems use only the fonts available on the user's computer, you may not be able to use specific fonts, such as Times New Roman, as some computers may not have those fonts installed. You can specify a font family, such as sans-serif, to ensure a font of a similar type is used on each computer. You can also specify multiple fonts, separated by commas, to allow the browser to display the first available font.

To set the font of a character

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Character Styles**, select the character style you want to modify.
4. On the **Properties** tab, click **Font**.
5. Specify the family, size, style, and other properties you want to modify. For more information about a property, click **Help**.

Adjusting the Space Around Characters

You can adjust the white space in all directions around characters by adjusting the margin and the padding. You can also adjust horizontal spacing by adjusting the kerning.

In terms of the CSS box model, modifying the padding property adjusts the space inside the border area. For example, if you create a border or background color for a character and you increase the size of the padding, the border moves away from the character.

Modifying the margin properties adjusts the space outside the border area. For example, if you create a border or background color for a character and you increase the size of the margins around the character, the border remains the same distance from the character. However, the position of the character changes since there is more white space between the modified character and other elements on the page.

To set the margin, padding, and kerning of a character

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Character Styles**, select the character style you want to modify.
4. On the **Properties** tab, click **Margin**.
5. Specify a value and select the unit of measure for the **Left**, **Right**, **Top**, and **Bottom** margin properties.
6. On the **Properties** tab, click **Padding**.
7. Specify a value and select the unit of measure for the **Left**, **Right**, **Top**, and **Bottom** padding properties.
8. On the **Properties** tab, click **Text**.
9. In the **Letter spacing** field, specify a value and select the unit of measure to adjust the kerning. For more information about a property, click **Help**.

Setting the Color and Other Characteristics of Characters

In general, ePublisher generates output based on the properties of the content from the original source documents. You can modify the appearance of your online content without modifying your source documents. The text properties allow you to modify several important characteristics of characters:

- Color
- Kerning between letters and words
- Underlining, overlining, and strikethrough text
- Vertical alignment to create subscripts and superscripts

To set the text characteristics of a character

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Character Styles**, select the character style you want to modify.
4. On the **Properties** tab, click **Text**.
5. Specify the appropriate values on the text-related properties, such as **Color** and **Letter spacing**. For more information about a property, click **Help**.

Modifying Characters for Bidirectional Languages

In some documents, such as those written with the Arabic or Hebrew script, and in some mixed-language contexts, text within a single paragraph may appear with mixed directionality. For example, some characters are read from left to right, while others within the paragraph may be read from right to left. This phenomenon is called **bidirectionality**, or **bid** for short.

If a document contains right-to-left characters, and if the browser can display the language with the proper character set, the browser must apply the bidirectional algorithm. If you prefer to control the handling of a particular phrase, you can apply a character style to that phrase and then define the character style with the **Direction** and **Unicode Bidi** properties.

To set the Unicode direction of a character

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Character Styles**, select the character style you want to modify.
4. On the **Properties** tab, click **Text**.
5. Under **Unicode**, set the appropriate values for the **Direction** and **Unicode Bidi** properties. For more information about these properties, click **Help**.

Defining the Appearance of Tables

Modifying the appearance of tables is different from modifying other elements in your content, such as paragraphs. The properties of tables are controlled by different layers. Some objects are in front of, or behind, other objects within the table. For example, the background property of the entire table is the first layer. The body, header, and footer properties reside in the next layer, which is on top of the background property. Paragraph properties are on top of that, followed by character properties, which are on the topmost layer.

With this model, you may have difficulty achieving the results you want when you try to adjust the appearance of a property for a table. You may need to experiment with the different property layers to fine-tune the appearance of tables. For example, if you try to create a transparent table by properly setting the body background property, but the table is not transparent, another layer may not be transparent. You need to make sure the background property is properly set for each layer in the table, including the table header, the paragraph, and the character styles.

Paragraph styles, such as CellHeading, CellBody, CellStep, and CellBullet give you additional control over formatting within tables in your generated output. When defining the left margin for a paragraph style used in a table, consider the size and weight of the font. For example, bold table headings need less pixels in their left margin than non-bold table text paragraphs so that both types of text align with each other.

The Prototype Style for Tables

The **Prototype** style is the parent to all other styles. When you set a property for the **Prototype** table style, other table styles inherit the value of that property. You can then override that value for specific styles as needed. The **Prototype** table style allows you to quickly change a default property and apply that change to all table styles within your Stationery project.

Setting the Background Color or Image of a Table

You can specify a color to use as the background of a table. You can also make the background transparent or specify an image to use as the background. In addition, you can specify alternative colors or images to alternate the appearance of rows or columns. Alternate shading of rows or columns is a useful layout to help minimize the number of lines and amount of information displayed, while organizing the data in a way that users can easily read and understand.

The background image for a table is behind other elements in a table. If you set a background color for the table, you cannot see the background image for the table. In addition, make sure the image is stored in the `Files` folder so it is part of the project.

To set the background color or image of a table

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Table Styles**, select the table style you want to modify.
4. On the **Properties** tab, click **Body Background**.
5. *If you want to specify a color for the background of a table*, select a color in the **Color** field, or type the RGB value of the color you want, such as `FFFFFF`. To make the table transparent, click the **Web** tab in the **Color** field, and then click **Transparent**.
6. *If you want to specify an image for the background of a table*, complete the following steps:
 - a. Save the image file in the `Files` folder for the project. ePublisher copies files from the `Files` folder when you generate the project.
 - b. In the **Image** field under **Background Image**, select the image you want to use. ePublisher lists only the image files stored in the `Files` folder.
 - c. Specify the tiling, scrolling, and position properties to position the image as you want it.
7. *If you want to alternate the appearance of rows or columns in the table*, specify the appropriate values for the **Alternate Information** properties. For more information about a property, click **Help**.

Setting the Border Style and Color of a Table

The border properties modify the appearance of the border that surrounds the outside of the table. However, some browsers display this information in different ways. For example, some browsers use the color and not the style of the border properties to define all other borders inside the table, unless that color has been previously defined.

For example, if you choose a red, dotted border for your table, the preview pane displays the outer edge of the table as a red, dotted border. All table cells inside the table have a red, solid border, unless the border properties for **Body** and **Header** (if applicable) are also defined. Not all browsers display the table the same way, so verify the results in multiple browsers when defining the table border and style.

To set the border style and color of a table

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Table Styles**, select the table style you want to modify.
4. On the **Properties** tab, click **Border**.
5. Click the tab for the side of the table you want to display a border, and then specify the color, style, and width for that border. For more information about a property, click **Help**.

Setting the Width and Height of a Table

You can define a fixed width and height for a table style. Make sure all the content of your tables will fit within the fixed height and width you specify. You can also use the table cell widths defined in your source documents.

To set the height and width of a table

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Table Styles**, select the table style you want to modify.
4. *If you want to define a fixed width and height for the table style*, complete the following steps:
 - a. On the **Properties** tab, click **HTML**.
 - b. Specify the appropriate values for the **Width** and **Height** properties. For more information about a property, click **Help**.
5. *If you want to use the cell widths defined in your source documents*, on the **Options** tab, set **Use document cell widths** to **Enabled**.

Setting the Vertical and Horizontal Alignment within a Table

You can specify the vertical and horizontal alignment of content within table cells. To define the vertical alignment, set the **Alignment** property for the table. To define the horizontal alignment, set the **Horizontal** alignment property for the paragraph style of the text in the table. These property values take effect unless a value is set at the paragraph style level, or a value is set for the table **Body**, **Header**, or **Footer** properties, if applicable.

To set the alignment of content within a table

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Table Styles**, select the table style you want to modify.
4. *If you want to define the vertical alignment*, complete the following steps:
 - a. On the **Properties** tab, click **Table**.
 - b. Specify the appropriate value in the **Vertical** field under **Alignment**.
5. *If you want to define the horizontal alignment*, complete the following steps:
 - a. In **Paragraph Styles**, select the paragraph style you want to modify.
 - b. On the **Properties** tab, click **Text**.
 - c. Specify the appropriate value in the **Horizontal** field under **Alignment**.

Adjusting the Space Within and Around a Table

After creating the external borders for your table, you can define the padding to specify the distance between the content and the borders within the table. This feature, which is enabled through CSS, is not supported by all browsers. View the output in several different browsers to verify your results.

You can also define the space around the table by adjusting the margin properties. Modifying the margin properties adjusts the space outside of the border area. For example, if you create a border for a table and you increase the size of the margins around the table, the border remains the same distance from the content of the table. However, the effective size of the table increases since there is more space between the table border and the other elements on the page.

To adjust the padding and margin of a table

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Table Styles**, select the table style you want to modify.
4. On the **Properties** tab, click **Table**.
5. Specify a value and select the unit of measure for the **Left**, **Right**, **Top**, and **Bottom** padding properties.
6. On the **Properties** tab, click **Margin**.
7. Specify a value and select the unit of measure for the **Left**, **Right**, **Top**, and **Bottom** margin properties.

Modifying Header, Footer, and Body Rows of a Table

In addition to modifying the appearance of an entire table, you can modify portions of a table, such as the header row, the footer row, and rows within the body of the table. To modify the appearance of a section of a table, you need to correctly define the parts of the table in your source documents. For example, in Microsoft Word, you need set the table property to define the header row. If you do not create a header row for a table in your source document, you cannot modify the header row appearance in your output. The first row of a table is not, by default, a header row.

To define the appearance of the header rows, specify the appropriate values for the **Header** and **Header background** properties for your table styles. These property values override the values specified for the **Table**, **Border**, and **Background** properties.

To define the appearance of the body rows, specify the appropriate values for the **Body** and **Body background** properties for your table styles. These property values override the values specified for the **Table**, **Border**, and **Background** properties.

To define the appearance of the footer rows, specify the appropriate values for the **Footer** and **Footer background** properties for your table styles. These property values override the values specified for the **Table**, **Border**, and **Background** properties.

Tables created in Microsoft Word do not identify footer rows. To control footer rows with Microsoft Word and ePubublisher, set up your table in your source document to reflect the desired appearance, or use footer paragraph styles in the footer row of the table. Then, ePubublisher can use the table set up from your source document and you can use the footer paragraph styles to modify the appearance as needed.

Tables created in Adobe FrameMaker identify the footer rows of tables, which allows you to specify the **Footer** and **Footer background** properties to modify footer rows for online delivery.

Modifying Cells of a Table

Modifying certain aspects of a table cell may also require you to modify the properties associated with the paragraphs that reside in the table cell. For example, if you make a table background color transparent, you may also need to modify the background color of the paragraphs in the table to make those cells transparent.

Cell spacing is used to create a transparent space between cells of a table. This capability allows you to create unique border structures, as the background color of a table can be seen through the space between table cells, which can have a different background color.

To adjust the spacing between cells

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Table Styles**, select the table style you want to modify.
4. On the **Properties** tab, click **Table**.
5. In the **Cell spacing** field, specify an appropriate value.

Defining the Appearance of Images

In general, ePublisher automatically transforms images from your source document to an optimized version for online distribution. However, if necessary, you can manually modify individual images or groups of images by applying a graphic style to the specific image or images you want to control. For example, you can define how images are resized, the maximum height and width for images, and the format for those images.

Supported Image Formats

ePublisher supports several image formats and rasterizes all other formats:

.jpg files

Provides a good format for photographs and images with many gradual variations in color. Although this format is compressed, the large number of colors can increase the size of the file when compared to the other supported image formats. The lossy data compression also reduces detail each time you save an image in this format. This format is not recommended for screen shots.

.gif files

Provides a good format for screen shots, since the number of colors can be reduced and the need for gradual variations in color is minimal. This format supports transparency and animation. However, this format supports a reduced color palette of 256 RGB colors (8-bit).

.png files

Provides a good multipurpose format for online presentation. This format supports up to 16 million RGB colors (24-bit) and uses lossless data compression.

.svg files

Provides a powerful format that is not supported in all output formats. Some browsers do not support `.svg` image files. If you use `.svg` image files, you need to configure the `.svg` options to specify whether to rasterize these images.

You can use **passthrough** conditions and coding to include any type of file within your final output, such as Flash `.swf` files. For example, you can create a paragraph style in your source documents that is not included in the print version of your documentation. You can apply a condition to this paragraph that has the **passthrough** setting selected in ePublisher. Then, this paragraph can provide the exact coding to include in your output. Be sure to include any referenced files in your `Files` folder of your project so ePublisher copies those files to your output location.

Image Quality and Processing

If ePublisher cannot use an original image in the output, or if ePublisher determines it needs to modify the image based on how it is included in the source document, ePublisher rasterizes the image using the options you define for your graphic styles in Style Designer. For example, you can define the dots per inch (DPI) and format for the final images. Rasterization of an image can cause the image to be blurry or distorted in the output.

For more information and specific considerations about your images, see “Image Formats and Considerations in FrameMaker” and “Image Styles and Considerations in Word”.

The Prototype Style for Images

The **Prototype** style is the parent to all other styles. When you set a property for the **Prototype** graphic style, other styles inherit the value of that property. The **Prototype** graphic style allows you to quickly change a default property and apply that change to all graphic styles within your Stationery project.

Defining Graphic Styles

All images are, by default, set to the **Prototype** style. If you need to modify any image properties, you can do so through the **Prototype** style. However, if you want to control a smaller set of images, even just one image, you need to assign a unique graphic style to those images.

To define a graphic style

1. Create a marker named **GraphicStyle** in your FrameMaker template.
2. Open your Stationery design project.
3. On the **View** menu, click **Style Designer**.
4. In **Graphic Styles**, define a set of graphic styles. Writers can use a marker or field code to specify the graphic style to apply to each image.

Setting the Border Style and Color of an Image

Borders are lines that can be drawn around any or all of the four sides of an image. In terms of the CSS box model, increasing the padding for a graphic style increases the space between the image and the border.

Not all browsers display border styles the same way. For example, some browsers may not differentiate dotted lines from solid lines. The size and spacing of the dots in a dotted line may also be different between various browsers and operating systems.

To set the border style and color of an image

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Graphic Styles**, select the graphic style you want to modify.
4. On the **Properties** tab, click **Border**.
5. Click the tab for the side of the image you want to display a border, and then specify the color, style, and width for that border. For more information about a property, click **Help**.

Modifying the Width, Height, and Positioning of an Image

ePublisher automatically transforms images in your source document into Web-ready formats. However, the size of your print image may not be appropriate for online delivery. ePublisher provides several ways to modify the size of your images for online delivery without affecting the original images.

If you know the exact dimensions you want to assign to an image, you can use the height and width properties of a graphic style. However, you are defining the dimensions of all images that use that graphic style. Unless you want all your images to be the same size, such as 150 pixels high and 275 pixels wide, this option is not the most effective way to modify the size of your images. For most situations, it is more efficient to define the maximum height and width for an image as opposed to assigning a fixed height and width. For more information, see “Setting the Maximum Width and Height for Images” and “Modifying Image Size by Scale”.

In most cases, you probably want to leave your images positioned as they are in your source documents. To visually enhance the layout and presentation of your online images, ePublisher allows you to set the position of any image according to CSS rules.

To set the width, height, and positioning of an image

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Graphic Styles**, select the graphic style you want to modify.
4. On the **Properties** tab, click **HTML**.
5. Specify the appropriate values for the width, height, and positioning properties.

Adjusting the Space Around Images

You can adjust the white space around images by adjusting the margin and the padding. In terms of the CSS box model, modifying the padding property adjusts the space inside the border area. For example, if you create a border or background color for an image and you increase the size of the padding, the border moves away from the image.

Modifying the margin properties adjusts the space outside the border area. For example, if you create a border or background color for an image and you increase the size of the margins around the image, the border remains the same distance from the image. However, the position of the image changes since there is more white space between the modified image and other elements on the page.

To set the margin and padding of an image

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Graphic Styles**, select the graphic style you want to modify.
4. On the **Properties** tab, click **Margin**.
5. Specify a value and select the unit of measure for the **Left**, **Right**, **Top**, and **Bottom** margin properties.
6. On the **Properties** tab, click **Padding**.
7. Specify a value and select the unit of measure for the **Left**, **Right**, **Top**, and **Bottom** padding properties.

Using Thumbnails for Images

ePublisher allows you to automatically reduce the size of images in your online content to thumbnails, which are reduced versions of your images. Users can view the full-sized image by clicking on the thumbnail. If you set the width and height values for thumbnails, ePublisher automatically creates the thumbnails you need and links them to the full-sized images. If you do not set a width and height for thumbnails, ePublisher inserts the image itself in your output and does not use thumbnails.

To use thumbnails for images

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Graphic Styles**, select the graphic style you want to modify.
5. On the **Options** tab, specify values, in pixels, for the **Maximum thumbnail width** and **Maximum thumbnail height** options.

Setting the Maximum Width and Height for Images

Most images in your content vary in width and height. For your online content, you need to ensure that all your images fit within the display area. In most cases, you have a maximum width or height constraint. For example, although some of your images may only be 250 pixels wide, you want to make sure that none of your images are wider than 275 pixels.

ePublisher allows you to set maximum width and height values for the size of your images. By modifying the size of your images in this way, you ensure that the resized images retain their original aspect ratio.

To set the maximum width and height of images

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Graphic Styles**, select the graphic style you want to modify.
5. On the **Options** tab, specify values, in pixels, for the **Maximum image width** and **Maximum image height** options.

Modifying Image Size by Scale

If you are not concerned with actual width and height of an image, you can specify a scaling percentage to apply to all images with the selected graphic style. This option allows you to modify the size of all images associated with the graphic style in relation to their original sizes. Modifying images in this way retains the original aspect ratio of each image.

To resize images based on a percentage

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Graphic Styles**, select the graphic style you want to modify.
5. On the **Options** tab, specify a percentage value for the **Scale %** option. For example, if you specify 50, each image with the selected graphic style will be reduced to half its original size.

Modifying Image Resolution

If the resolution of the images in your source documents is set for print, such as a resolution of 300 dots per inch (dpi) or higher, you may want to reduce the image resolution for online delivery. High dpi images create larger files that require more time to download. In addition, most monitors display a resolution of 96 dpi, so higher resolutions do not increase the quality of the image displayed online.

Although transforming an image from 300 dpi to 96 dpi helps optimize your images for online delivery, the width and height of your images is also affected. Because a resolution of 300 dpi contains more dots per inch than a resolution of 96 dpi, when ePublisher transforms the image, the image will be roughly 68% smaller than the original image. For example, a 300 dpi image that is 100 x 100 pixels will be 32 x 32 pixels when transformed to a 96 dpi image.

To counter this effect, use the **Scale %** option in conjunction with the **Render DPI** option to control the size of your images. In the example of transforming a 300 dpi image to 96 dpi, set the **Scale %** option to **312**, which then generates an image that has roughly the same dimensions as the original source image. You can also use the **Scale %** and **Render DPI** options together with the **Maximum image height** and **Maximum image width** options to make sure your images are correctly sized for online delivery.

To set the image resolution (dpi)

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Graphic Styles**, select the graphic style you want to modify.
5. On the **Options** tab, specify a value in dpi for the **Render DPI** option.

Setting Color Bit Depth

The range of colors available through digital computer images is usually expressed in terms of bit depth. This expression refers to the number of bits of data carrying the color information. Common bit depth levels for images include 8-bit and 24-bit color. In general, more bits of data make more colors available. The **Color bit depth** option applies only to .gif and .png images. .jpg images always generate 32-bit images.

To set the color-bit depth for .gif and .png images

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Graphic Styles**, select the graphic style you want to modify.
5. On the **Options** tab, select a value for the **Color bit depth** option.

Choosing an Image File Format and Quality Level

By default, if ePublisher needs to rasterize an image, it transforms the image in your source documents, regardless of what file format it is, to a Web-ready .jpg image. In some instances, you may want to use other image formats for online delivery. For example, .gif images can produce similar quality images as .jpg, but the file size is smaller. .gif images can also support transparent colors. You can also create .png images, which combine some of the better qualities of both .jpg and .gif.

If you select **JPG** in the **Format** options for a graphic style, you can specify the quality of the images. The quality level impacts both the visual aspects of your images and the size of the generated files. Higher-quality images require larger files, which require more time to download and display. The **JPG Quality** option does not affect .gif or .png images.

To choose the file format and quality for online images

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Graphic Styles**, select the graphic style you want to modify.
5. On the **Options** tab, select a value for the **Format** option.
6. In the **Output file extension** field, specify the proper file extension based on the value you selected for the **Format** option, such as .jpg, .gif, or .png.
7. *If you selected JPG in the Format option*, in the **JPG Quality** field, specify the percentage value you want for your online images. A value of 100 creates the highest quality image that mimics your original image.

Creating Grayscale Images

ePublisher allows you to transform your original color images into grayscale versions for online viewing. If you enable the **Grayscale** option, ePublisher removes the color saturation of the original images when those images are transformed with your online content. ePublisher displays the color versions of your images in the preview pane to generate the preview more quickly.

To create grayscale images

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Graphic Styles**, select the graphic style you want to modify.
5. On the **Options** tab, set the **Grayscale** option to **Enabled**.

Setting Transparency for .gif and .png Images

In some images, you may want to set a color to transparent. For example, if your source document has a white background, images with a white background appear as though they do not have a background. However, if your online content has a different color background, the background of these images appear as white areas. You can enable the transparency option in ePublisher to transform the white background into a transparent one. Only .gif and .png images support transparent colors.

Note: Some browser versions do not support transparent colors, especially in .png images.

To set transparency for .gif and .png images

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Graphic Styles**, select the graphic style you want to modify.
5. On the **Options** tab, set the **Transparent** option to **Enabled**.

Defining the Appearance of Pages

The way you present your online content greatly impacts the usability and readability of that content in the online environment. Information layout and design for print and online content are usually very different. ePublisher allows you to format your source documents to optimize for printed delivery, and then use ePublisher to deliver the content optimized for online presentation.

Unlike printed content, online documentation can include extra navigation features, follow different rules regarding topic length and page breaks, and include unique page layouts for specific types of information. While each output format varies in the types of files that are delivered, most output formats are based on HTML. Therefore, when you decide how to present your content online, consider the layout of your content on an HTML page.

In ePublisher, the overall appearance of each topic of your output is controlled by page styles. You can modify page style properties with Style Designer to define whether to include navigation browse buttons on each page, whether to include company logo and contact information, and other design elements for each page that uses a specific page style. To deliver your online content as individual chunks, or pages of information, define page breaks based on paragraph styles, such as headings. For more information, see “Defining New Pages (Page Breaks)”.

The Prototype Style for Pages

The **Prototype** style is the parent to all other styles. When you set a property for the **Prototype** page style, other page styles inherit the value of that property. You can then override that value for specific styles as needed. The **Prototype** page style allows you to quickly change a default property and apply that change to all page styles within your Stationery project.

Displaying Company Logo and Information on a Page

Most HTML-based output formats support adding company information, such as company name, email address, and company logo, as part of a page. Once you specify your company information in the **Target Settings** for your project, you can select different locations in which to place the content on each page. If you specify a value for **Company webpage**, ePublisher links the specified company name to the specified Web page. If you specify **Company email address**, ePublisher creates a `mailto:` link to the specified address.

If you add a logo image to your project, ePublisher displays the logo next to your company contact information on the top or the bottom of your output pages. To include a logo, you must first store the image file in the `Files` folder of your project.

To specify your company information and the location on the page

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. Specify the appropriate values for the **Company Information** settings, such as **Company name** and **Company phone number**. If you do not see these fields in the list of target settings, the output format for the active target does not support this feature. For more information about a setting, click **Help**.
4. Click **OK** to save the target settings.
5. On the **View** menu, click **Style Designer**.
6. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
7. In **Page Styles**, select the page style you want to modify.
8. On the **Options** tab, specify the appropriate values for the **Company info...** options. For more information about an option, click **Help**.

Note: Company information is not displayed in the Preview pane.

Modifying the Appearance of the Company Information

You can modify the appearance of the company information by creating an override for the `Page.asp` file. This file controls the content coding for each page of your content. When you modify this file, you need to change the appropriate part of the file based on whether you want to modify the appearance of the company information at the top of the page or at the bottom of the page. You can also modify many other aspects of your content pages. For more information, see “Customizing the Content Page Design”.

To modify the appearance of the company information

1. Create an override file for the `Page.asp` file by copying it from the installation folder to your format or target override folder in your project, based on whether you want to modify the appearance of the company information for an output format or for a specific target. For more information, see “Creating Format Overrides” and “Creating Target Overrides”.
2. Open the `Page.asp` file you copied to the `Formats\formatname\Pages` or `Targets\targetname\Pages` folder in your project.
3. Locate the code at the top or bottom of the `Page.asp` file, based on whether you want to change the appearance of the information at the top or the bottom of the page.

- To modify the top of the page, find the following line of code:

```
<table wpage:condition="company-info-top" align="left" wpage:attribute-align="company-info-top-alignment">
```

- To modify the bottom of the page, find the following line of code:

```
<table wpage:condition="company-info-bottom" align="right" wpage:attribute-align="company-info-bottom-alignment">
```

4. ***If you want to modify the appearance of the company name***, create an override for `webworks.css` from the `Formats\[your format]\Pages\css` installation directory. Then, find `td.WebWorks_Company_Name_Top` in `Pages\css\webworks.css`, and modify the CSS attributes, such as the font-size or font-family, and required.
5. ***If you want to modify the appearance of the company phone number***, find the following code immediately after the line of code you found at the top or bottom of the file:

```
<tr wpage:condition="company-phone-exists">
```

```
<td class="WebWorks_Company_Phone_Bottom">
```

```
or <td class="WebWorks_Company_Phone_Top">
```

Look for `td.WebWorks_Company_Phone_Bottom` or `td.WebWorks_Company_Phone_Top` in `Pages\css\webworks.css` and change the font-size or font-family

6. ***If you want to modify the appearance of the company fax number***, find the following code immediately after the line of code you found at the top or bottom of the file:

```
<tr wpage:condition="company-fax-exists">
```

```
<td class="WebWorks_Company_Fax_Bottom">
```

```
or <td class="WebWorks_Company_Fax_Top">
```

Look for `td.WebWorks_Company_Fax_Bottom` or `td.WebWorks_Company_Fax_Top` in `Pages\css\webworks.css` and change the font-size or font-family

7. ***If you want to modify the appearance of the company email address***, find the following code immediately after the line of code you found at the top or bottom of the file:

```
<tr wwpage:condition="company-email-exists">
```

```
<td class="WebWorks_Company_Email_Bottom">
```

```
or <td class="WebWorks_Company_Email_Top">
```

Look for `td.WebWorks_Company_Email_Bottom` or `td.WebWorks_Company_Email_Top` **in** `Pages\css\webworks.css` **and change the** `font-size` **or** `font-family`

Setting the Background Color or Image of a Page

You can specify a color to use as the background of a page. You can also specify an image to use as the background. This capability allows you to include a watermark, such as `DRAFT` for initial internal versions of your online content. The background image for a page is behind other elements on the page. If you set a background color for the page, you cannot see the background image for the page. Make sure the image is stored in the `Files` folder for the project so the image file is copied to your output folder. Also make sure the image does not make the text too difficult to read.

To set the background color or image of a page

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Page Styles**, select the page style you want to modify.
4. On the **Properties** tab, click **Background**.
5. *If you want to specify a color for the background of a page*, select a color in the **Color** field, or type the RGB value of the color you want, such as `FFFFFF`.
6. *If you want to specify an image for the background of a page*, complete the following steps:
 - a. Save the image file in the `Files` folder for the project. ePublisher copies files from the `Files` folder when you generate the project.
 - b. In the **Image** field under **Background Image**, select the image you want to use. ePublisher lists only the image files stored in the `Files` folder.
 - c. Specify the tiling, scrolling, and position properties to position the image. For more information about a property, click **Help**.

Setting the Border Style and Color of a Page

Borders are lines that can be drawn around any or all of the four sides of a page. In terms of the CSS box model, increasing the padding for a page style increases the space between the content within the page and the border of the page.

Not all browsers display border styles the same way. For example, some browsers may not differentiate dotted lines from solid lines. The size and spacing of the dots in a dotted line may also be different between various browsers and operating systems. Some browsers may not display page borders at all. If you want a page border, view the generated output in multiple browsers to verify that your settings create the appearance you want.

To set the border style and color for a page

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Page Styles**, select the page style you want to modify.
4. On the **Properties** tab, click **Border**.
5. Click the tab for the side of the page you want to display a border, and then specify the color, style, and width for that border. For more information about a property, click **Help**.

Adjusting the Space Around a Page

You can adjust the white space around a page by adjusting the margin and the padding. In terms of the CSS box model, modifying the padding property adjusts the space inside the border area. For example, if you create a border or background color for a page and you increase the size of the padding, the border moves away from the content in the page.

Modifying the margin properties adjusts the space outside the border area. For example, if you create a border or background color for a page and you increase the size of the margins around the page, the border remains the same distance from the content in the page. However, the position of the content changes because the space outside the border increases.

To set the margin and padding of a paragraph

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Page Styles**, select the page style you want to modify.
4. On the **Properties** tab, click **Margin**.
5. Specify a value and select the unit of measure for the **Left**, **Right**, **Top**, and **Bottom** margin properties.
6. On the **Properties** tab, click **Padding**.
7. Specify a value and select the unit of measure for the **Left**, **Right**, **Top**, and **Bottom** padding properties.

Using a Custom CSS to Modify the Appearance of Content

You can use a custom CSS file to modify the appearance of your content instead of using Style Designer for HTML-based output formats. By using style names in your source document that match the classes defined in your custom CSS file, such as **div.Heading1**, you can make sure your content uses your custom CSS file. You do not need to create any other association between styles in your source documents and the styles in Style Designer. Make sure you define each tag and class from your generated output in your custom CSS file.

To use a custom CSS file with your project

1. Store your custom CSS file in the `Files` folder in your project.
2. Open your Stationery design project.
3. On the **View** menu, click **Style Designer**.
4. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
5. In **Page Styles**, select the page style you want to modify.
6. On the **Options** tab, select your custom CSS file in **Custom document css file**.

Modifying the Location and Separators of Breadcrumbs

Breadcrumbs form a linked path to show users the location of the current topic in your online content. This clickable path steps you through the topics in the hierarchy above the current topic. You can display breadcrumbs at the top of the page, at the bottom of the page, or both. The breadcrumb trail at the top of the output page is enabled by default.

To modify the location of the breadcrumb trail

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Page Styles**, select the page style you want to modify.
5. On the **Options** tab, select the appropriate value for the **Breadcrumbs shown at top of page** and **Breadcrumbs shown at bottom of page** options.
6. On the **Properties** tab, click **Navigation**.
7. Specify the appropriate values for the **Breadcrumbs Separator** and **Alignment** properties. For more information about a property, click **Help**.

Modifying the Appearance of Breadcrumbs

Breadcrumbs form a linked path to show users the location of the current topic in your online content. If you enabled breadcrumbs for your output, you have several options to define the appearance of the breadcrumbs. For more information, see “Modifying the Location and Separators of Breadcrumbs”.

For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.

To modify the appearance of the breadcrumb trail

1. *If you want to override the CSS settings for an output format*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Format Override Directory**.
 - b. Create the `Formats\formattype\Pages\css` folder in your project folder, where *formattype* is the name of the output format you want to override, such as `Microsoft HTML Help 1.x`.
2. *If you want to override the CSS settings for a target*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Target Override Directory**.
 - b. Create the `Targets\targetname\Pages\css` folder in your project folder, where *targetname* is the name of the target you want to override.
3. Copy the `webworks.css` file from the following folder to the override folder you created within your project folder:
`Program Files\WebWorks\ePublisher Designer\Formats\formattype\Pages\css`
4. Open the `webworks.css` file you copied to your project override folder.
5. Find the code for `div.WebWorks_Breadcrumbs` and modify the values specified within the braces:
 - To modify the text color, specify the desired RGB color value for the `color` option.
 - To modify the font, specify the name of the font you want for the `font-family` option.
 - To modify the size of the font, specify the size you want for the `font-size` option.
6. Save the `webworks.css` file.
7. Regenerate your project to review the changes.

Choosing the Location of Navigation Browse Buttons

If you have included navigation buttons in your output, you can specify whether to display the browse buttons at the top or the bottom of each page. You can also specify whether to align the button along the right or left side. Not all output formats support navigation browse buttons.

To modify the location of navigation browse buttons





1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Page Styles**, select the page style you want to modify.
5. On the **Options** tab, select the appropriate value for the **Navigation links shown at top of page** and **Navigation links shown at bottom of page** options.
6. On the **Properties** tab, click **Navigation**.
7. Specify the appropriate values for the **Navigation Alignment** properties. For more information about a property, click **Help**.

Modifying the Navigation Browse Buttons

You can use customized navigation browse buttons in your online content. For more information about customizing the WebWorks Help navigation buttons, see “Customizing the Toolbar in WebWorks Help”.

To change the navigation button images

1. *If you want to override the images for an output format*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Format Override Directory**.
 - b. Create the `Formats\formattype\Pages\Images` folder in your project folder, where *formattype* is the name of the output format you want to override, such as `Dynamic HTML`.
2. *If you want to override the images for a target*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Target Override Directory**.
 - b. Create the `Targets\targetname\Pages\Images` folder in your project folder, where *targetname* is the name of the target you want to override.
3. Paste the `.gif` files you want to use with names identical to those you want to replace in the `Images` folder you created. The following table lists the default button images and their file names and sizes.

Image	Name	Image Size	Description
	prev.gif	Width: 0.722 inch Height: 0.333 inch	Sends the user to the previous HTML document.
	prevx.gif	Width: 0.722 inch Height: 0.333 inch	Displayed when there is no previous HTML document available.
	next.gif	Width: 0.722 inch Height: 0.333 inch	Sends the user to the next HTML document.
	nextx.gif	Width: 0.722 inch Height: 0.333 inch	Displayed when there is no next HTML document available.

4. Regenerate your project to review the changes.

Associating a Page with a Page Style

By default, output is associated with the **Prototype** page style in ePublisher. Therefore, you do not have to do anything to associate pages with the **Prototype** page style. However, if you want to have more than one page layout, you need to use additional page styles.

To associate a page with a page style

1. Define page breaks based on paragraph styles, such as heading styles.
2. Define additional page styles in the Stationery.
3. *If you want to associate a page with a page style other than the Prototype page style*, add a PageStyle marker or field code in your source document after the new page style, such as the heading that starts the page. The PageStyle marker or field code identifies what page style to use for that page.

Provide writers with the defined page style names they need to specify in their PageStyle markers or field codes.

Defining the Appearance of Links

You can customize the appearance of links, also known as hyperlinks and hypertext links. You can define the color and text decoration, such as underline, for the currently selected link (active link), links you have previously visited, links you hover over, and the links you have not previously visited. These link settings are stored in the `webworks.css` file. You can create an override for the `webworks.css` file and then modify the tags that define the links.

Note: These changes do not affect output formats, such as Simple HTML, that do not use cascading style sheets.

In your source files, apply a character style to identify the text to include in each link. The character style can also affect the appearance of the link.

To change the color of links

1. In your Stationery design project, go to **Advanced** menu bar item and either click **Manage Format Customizations** or **Manage Target Customizations**.
2. Navigate to `Pages\css` in this window.
3. Double click the `webworks.css` file and it will be bolded to indicate that a customization has been created.
4. Double click again, and the CSS file should be opened in a text editor:
5. Find the following code and modify the values specified within the braces:

```
a:link:active { color: #0000CC }  
a:link:hover { color: #CC0033 }  
a:link { color: #3366CC }  
a:visited { color: #9999CC }
```

- To modify the text color, specify the desired RGB color value for the `color` option of the appropriate link type and state.
 - To add text decoration, such as an underline, add the `text-decoration: underline;` option to the type of link you want to have an underline, and add the `text-decoration: none;` option to the other types of link.
 - To make the link text bold when you hover over the link, add the `font-weight: bold;` option to the `a:link:hover` definition.
6. Save the `webworks.css` file.
 7. Save and close your project.
 8. Reopen your project and regenerate to review the changes.

Saving a Snapshot (Backup Copy) of Your Project

As you continue to make changes to your Stationery design project, you may want to revert to a previous state, before you implemented a specific feature. You can save a backup copy of your project and then use this backup copy at a later time, if needed.

As you develop your Stationery design project, periodically save a snapshot of your project. You can use this snapshot to revert to your Stationery design project at a specific point in time. This snapshot can also help you if your design project or Stationery become corrupted in the future. For more information about the specific files and folders, see “Backing Up Your Stationery Design Project, Stationery, and Projects”.

To save a snapshot (backup copy) of your project

1. Save your project and close ePublisher.
2. Copy your project folder and all its subfolders and files, and your source documents folder, maintaining the same structure, to another location. For example, create a folder with the date from today as the name. Then, copy your project folder and your source documents folder into the folder you created. This copy is your snapshot.

Defining the Processing of Markers and Field Codes

If you want to create your own field codes and markers, you can define the function of each field code and marker you create. ePublisher provides many default field codes and markers to provide various built-in features and capabilities. For more information about which output formats support each feature, see “Features Available in Each Output Format”.

Default Marker Name	Description
AbbreviationTitle	Specifies alt text for an abbreviation, such as SS#. This text is displayed when a user hovers over the abbreviation in the output.
AcronymTitle	Specifies alt text for an acronym, which is displayed when a user hovers over the acronym in the output.
Citation	Specifies a citation as alt text for a quote, which is displayed when a user hovers over the citation in the output.
Context Plugin	Specifies context plug-ins for Eclipse help systems. Other Eclipse plug-ins can use the context plug-in IDs to call the Eclipse help system. For more information, see “Using Markers to Specify Context Plug-ins in Eclipse Help”.
DropDownEnd	Marks the end of an expand/collapse section. For more information, see “Defining Expand/Collapse Sections (Drop-Down Hotspots)”.
Filename	Specifies the name of an output file for a page or an image. For more information, see “Defining File Names”.
GraphicScale	Specifies a percentage to resize an image, such as 50 or 75 percent.
GraphicStyle	Specifies the name of a graphic style defined in the project to apply to an image. This marker is an internal marker name that is not displayed in Stationery Designer. You cannot create a marker with a different name and assign it this functionality. For more information, see “Defining the Appearance of Images”.
ImageAltText	Specifies alt text for an image. This text is added to the <code>alt</code> attribute of the <code>img</code> tag in the output.
ImageAreaAltText	Specifies alt text for clickable regions in an image map. This text is displayed when a user hovers over the region in the output.
ImageLongDescByRef	Specifies the path to the file that contains the long description for an image. This description is used when you create accessible content.
ImageLongDescNotReq	Specifies that a long description is not required for an image, which bypasses this accessibility check for that image.
ImageLongDescText	Specifies the long description for an image. This description is used when you create accessible content.
Keywords	Specifies the keywords to include in the <code>META</code> tag for the topic. This <code>META</code> tag improves searchability on the Web.
PageStyle	Specifies the name of a page style defined in the project to apply to a topic. This marker is an internal marker name that is not displayed in Stationery Designer. You cannot create a marker with a different name and assign it this functionality. For more information, see “Defining the Appearance of Pages”.

Default Marker Name	Description
PassThrough	Specifies that ePublisher place the contents of the marker directly into the generated output without processing the content in any way. For example, you could use a PassThrough marker if you wanted to embed HTML code within your generated output.
Popup	Specifies the start of the content to include in a popup window. The content is displayed in a popup window when you hover over the link. When you click the link in some output formats, the topic where the popup content is stored, such as the glossary, is displayed. For more information, see “Defining Popup Windows”.
PopupEnd	Marks the end of the content to include in a popup window.
PopupOnly	Specifies the start of the content to include in a popup window. The content is displayed in a popup window when you hover over or click the link.
RubiComposite	No longer supported.
SeeAlsoKeyword	Specifies an internal identifier for a topic. SeeAlsoLink markers in other topics can list this identifier to create a link to this topic. For more information, see “Defining See Also Links”.
SeeAlsoLink	Identifies an internal identifier from another topic to include in the list of see also links in this topic.
SeeAlsoLinkDisplayType	Specifies whether to display the target topics on a popup menu or in a window. By default, the links are displayed in the Topics Found window. To display a popup menu, set the value to <code>menu</code> . This marker is supported only in HTML Help.
SeeAlsoLinkWindowType	Specifies the name of the window defined in the hhp file, such as TriPane or Main, that the topic opens in when the user clicks the link. This marker is supported only in HTML Help.
TableStyle	Specifies the name of a table style defined in the project to apply to a table in versions of Microsoft Word that did not support table styles. This marker is an internal marker name that is not displayed in Stationery Designer. You cannot create a marker with a different name and assign it this functionality.
TableSummary	Specifies a summary for a table, which is used when you create accessible content.
TableSummaryNotReq	Specifies that a summary is not required for a table, which bypasses this accessibility check for that table.
TOCIconHTMLHelp	Identifies the image to use as the table of contents icon for this topic in the HTML Help output format.
TOCIconJavaHelp	Identifies the image to use as the table of contents icon for this topic in the Sun JavaHelp output format.
TOCIconOracleHelp	Identifies the image to use as the table of contents icon for this topic in the Oracle Help output format.
TOCIconWWHelp	Identifies the image to use as the table of contents icon for this topic in the WebWorks Help output format.
TopicAlias	Specifies an internal identifier for a topic that can be used to create a context-sensitive link to that topic. For more information, see “Defining Context-Sensitive Help Links”.
TopicDescription	Specifies a topic description for a context-sensitive help topic in Eclipse help systems. For more information, see “Using Markers to Specify Topic Descriptions for Context-Sensitive Help Topics in Eclipse Help”.
WhatIsThisID	Identifies a What’s This help internal identifier for creating context-sensitive What’s This field-level help in the HTML Help output format.

Default Marker Name	Description
WikiCategory	Specifies the category or label you want to assign to a topic when generating Wiki output. For more information, see “Defining Wiki Categories or Labels”.
WindowType	Specifies the name of the window defined in the help project that the topic should be displayed in. In HTML Help, the window names are defined in the hhp file. This marker is supported in HTML Help and Oracle Help.

Defining File Names

By default, ePublisher automatically assigns file names to your generated output files for topics (pages) and images (graphics). ePublisher assigns output file names using a default naming rule. You can customize this naming convention using one of the following methods:

- Specifying a different topic (page) or image (graphic) naming pattern for ePublisher to use in the target settings for your output.
- Inserting Filename markers into source documents that specify the topic (page) or image (graphic) file name ePublisher should use for the file when generating output.

For more information about which output formats support this feature, see “Features Available in Each Output Format”.

Specifying File Names for Pages Using Page Naming Patterns

By default, ePublisher uses the following values when specifying file names for pages:

\$D;.\$DN;.\$PN

This specifies that ePublisher name the files using the following syntax when it generates page files for a target:

SourceDocumentName.SourceDocumentNumber.TopicNumber

The parts of the default naming rule are defined as follows:

SourceDocumentName

Identifies the name of the source document that the topic came from without the file extension.

SourceDocumentNumber

Identifies the number of the source document in the order it is included in its containing group in the project, such as 1 for the first source document in a group and 2 for the second source document in a group. This value starts at 1 for the first source document in each group in your project.

TopicNumber

Identifies the number of the topic (output page) generated from the source document, such as 1 for the first topic generated from a source document and 2 for the second topic generated from a source document. This value starts at 1 for the first topic in each source document.

For example, if you have a `MyFile.fm` source document that contains three topics that start with a paragraph style that has a page break priority set in Style Designer, ePublisher generates the following default output file names: `MyFile.1.1.html`, `MyFile.1.2.html`, and `MyFile.1.3.html`. You can change the default file extension for each page style.

You can use the following values to specify a page file naming pattern for a target:

Note: Each value you specify must begin with a dollar sign (\$) character and end with a semicolon (;) character. Inserting a period (.) character immediately before the value specifies that ePublisher use a period to separate the values when generating output.

Value	Description
\$P;	Includes the name of the project in the file name.
\$T;	Includes the name of the target in the file name.
\$G;	Includes the name of the group in Document Manager that contains the file name.
\$C;	Includes the project to project linking context value of the group in the file name. WebWorks Help and WebWorks Reverb use the context value when generating merged, or multivolume help that includes context-sensitive help. In WebWorks Help/Reverb, you need to include this context and the TopicAlias value in the help call to display the correct help topic. For more information, see “Merging Help Systems (Multivolume Help)” and “Opening Context-Sensitive Help in WebWorks Help using Standard URLs”.
\$H;	Includes the heading text or title of the topic in the file name.
\$D;	Includes the name of the source document that the topic came from in the file name.

Value	Description
\$DN;	Includes the source document number in the file name. The source document number is the number that identifies the position of the source document in the project.
\$PN;	Includes the page number in the file name. The page number is the number of the page that the topic is on in the source document.

You can also specify if you want ePublisher to convert spaces in file names to underscore (`_`) characters when generating output. If you enable this setting, when you generate output, spaces in file names are replaced with the underscore character. For example, with this setting disabled, file names display as `Word1 Word2.html`. With this setting enabled, when you generate output file names display as `Word1_Word2.html`.

To specify a page file naming patterns for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target** menu, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **Files**, specify the appropriate values for the page file naming pattern you want to use. For more information about file settings and values, click **Help**.

Specifying File Names for Images Using Graphic Naming Patterns

ePublisher preserves the original file names for images imported by reference. If images are inserted directly into a source document, or if ePublisher cannot process the image by reference, then ePublisher assigns a file name using a graphic naming pattern.

By default ePublisher uses the following values when specifying file names for images:

\$D;\$.DN;\$.PN;\$.GN

This specifies that ePublisher name the files using the following syntax when it generates image files for a target:

SourceDocumentName.SourceDocumentNumber.TopicNumber.ImageNumber

The parts of the default naming rule are defined as follows:

SourceDocumentName

Identifies the name of the source document that the topic came from without the file extension.

SourceDocumentNumber

Identifies the number of the source document in the order it is included in its containing group in the project, such as 1 for the first source document in a group and 2 for the second source document in a group. This value starts at 1 for the first source document in each group in your project.

TopicNumber

Identifies the number of the topic (output page) generated from the source document, such as 1 for the first topic generated from a source document and 2 for the second topic generated from a source document. This value starts at 1 for the first topic in each source document.

ImageNumber

Identifies the number of the image in the topic generated from the source document, such as 1 for the first image generated in a topic and 2 for the second image generated in a topic. This value starts at 1 for the first image in each topic.

For example, if you have a source document named `MyFile.doc` that contains two images in the first topic that generates an output file, and three images in the second topic that generates an output file, and all the images are directly included in the source document, ePublisher generates the following default output image file names:

`MyFile.1.1.1.jpg`, `MyFile.1.1.2.jpg`, `MyFile.1.2.1.jpg`, `MyFile.1.2.2.jpg`, and `MyFile.1.2.3.jpg`. You can change the default format and file extension for each graphic style.

You can use the following values to specify an image (graphic) file naming pattern for a target:

Note: Each value you specify must begin with a dollar sign (\$) character and end with a semicolon (;) character. Inserting a period (.) character immediately before the value specifies that ePublisher use a period to separate the values when generating output.

Value	Description
\$P;	Includes the name of the project in the file name.
\$T;	Includes the name of the target in the file name.
\$G;	Includes the name of the group in Document Manager that contains the file name.
\$C;	Includes the project to project linking context value of the group in the file name. WebWorks Help and WebWorks Reverb use the context value when generating merged, or multivolume help that

Value	Description
	includes context-sensitive help. In WebWorks Help/Reverb, you need to include this context and the TopicAlias value in the help call to display the correct help topic. For more information, see “Merging Help Systems (Multivolume Help)” and “Opening Context-Sensitive Help in WebWorks Help using Standard URLs”.
\$H;	Includes the heading text or title of the topic in the file name.
\$D;	Includes the name of the source document that the topic came from in the file name.
\$DN;	Includes the source document number in the file name. The source document number is the number that identifies the position of the source document in the project.
\$PN;	Includes the page number in the file name. The page number is the number of the page that the topic is on in the source document.
\$GN;	Includes the graphic number in the file name. The graphic number is the sequential number of the graphic in the generated output, and it is based on the position of the graphic in the generated output.

You can also specify if you want ePublisher to convert spaces in file names to underscore (`_`) characters when generating output. If you enable this setting, when you generate output, spaces in file names are replaced with the underscore character. For example, with this setting disabled, file names display as `Word1 Word2.jpg`. With this setting enabled, when you generate output file names display as `Word1_Word2.html`.

To specify an image (graphic) file naming pattern for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target** menu, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **Files**, specify the appropriate values for the graphic file naming pattern you want to use.

Using Markers to Define File Names

You can use markers and field codes to specify the file name for a topic or image. Writers insert a Filename marker into their source documents and specify the file name they want to assign to the topic or image in the marker.

The default name for this marker type or field code is Filename. You can define your own marker style with a different name in your Stationery and assign the Filename marker type to it. Then, writers can use this marker type or field code to specify the output file name for each topic and included image.

Note: ePublisher uses page and graphic naming patterns to assign file names to all topics and images that do not include a Filename marker type or field code. For more information, see “Specifying File Names for Pages Using Page Naming Patterns” and “Specifying File Names for Images Using Graphic Naming Patterns”.

To assign file name behavior to file name markers

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Marker Styles**, select the marker style you want to modify.
4. On the **Options** tab, set **Marker type** to **Filename**.

Defining Context-Sensitive Help Links

Context-sensitive help links provide content based on the context of what the user is doing. In many cases, this help content is based on the window that is open and active. For example, the **Help** button on a window in a software product can open a specific help topic that provides important information about the window:

- What the window allows you to do
- Brief concepts needed to understand the window
- Guidance for how to use the window
- Descriptions about each field on the window, valid values, and related fields
- Links to related topics, such as concepts and tasks related to the window

The help topic can also be embedded in the window itself, such as an HTML pane that displays the content of the help topic. Providing this content when and where the user needs it, without requiring the user to search through the help, keeps the user productive and focused. This type of help also makes the product more intuitive by providing answers when and where needed.

There are several methods for creating context-sensitive help. In addition, output formats use different mechanisms to support context-sensitive help. You can reference a topic in the following ways:

File name

Use a Filename marker to assign a file name to a topic. Each topic can have no more than one Filename marker by default. However, you can create a custom mapping mechanism using file names. Then, you can open the specific topic with that file name. However, if your file naming changes, you need to change the link to the topic. This file naming approach delivers context-sensitive help capabilities in output formats that do not provide a mapping mechanism.

Internal identifier (topic alias)

Use a TopicAlias marker to define an internal identifier for each topic. The benefit of using an internal identifier is that it allows file names to change without impacting the links from the product. The writer inserts this marker in a topic and specifies a unique value for that topic. Then, the mapping mechanism of your output format determines how that internal identifier is supported. Some output formats, such as HTML Help, use a mapping file that defines these topic aliases. You can create more than one TopicAlias marker in a topic to allow multiple context-sensitive links to display the same topic.

To simplify the coding of your source documents, you can use the same marker to define both the file name and the topic alias for each topic file. In Style Designer, set the **Marker type** option for the marker you want to use to **Filename and topic alias**. For more information about which output formats support this feature, see “Features Available in Each Output Format”.

For more information about using markers to enable context-sensitive help links, see the following topics:

- “Defining Filename Markers for Context-Sensitive Help Links”
- “Defining Topic Alias Markers for Context-Sensitive Help Links”

For more output format-specific information about using and customizing context-sensitive help, see the following topics:

- “Using Context-Sensitive Help in WebWorks Help”
- “Using Context-Sensitive Help in HTML Help”
- “Using Context-Sensitive Help in Oracle Help and Sun JavaHelp”

Defining Filename Markers for Context-Sensitive Help Links

To enable context-sensitive help links using file names, you need to enable the Filename marker. By default, ePublisher sets the **Marker type** option for a marker named Filename to **Filename**. You can create a marker with a different name and set the **Marker type** option for that marker to **Filename**.

Then, writers can use this marker in the source documents to define a file name for each topic that will be opened by the application. File names must follow these guidelines:

- Must be unique
- Can only contain characters valid for file names

To assign file name behavior to file name markers

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Marker Styles**, select the marker style you want to modify.
4. On the **Options** tab, set **Marker type** to **Filename**.

Defining TopicAlias Markers for Context-Sensitive Help Links

To enable context-sensitive help links using topic IDs, you need to enable the TopicAlias marker. By default, ePublisher sets the **Marker type** option for a marker named TopicAlias to **Topic alias**. You can create a marker with a different name and set the **Marker type** option for that marker to **Topic alias**.

Then, writers can use this marker in the source documents to define a topic ID in each topic that will be opened by the application. Topic IDs must follow these guidelines:

- Must be unique
- Must begin with an alphabetical character
- May contain alphanumeric characters
- May not contain special characters or spaces, with the exception of underscores (_)

To assign topic alias behavior to topic alias markers

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In **Marker Styles**, select the marker style you want to modify.
4. On the **Options** tab, set **Marker type** to **Topic alias**.

Defining Expand/Collapse Sections (Drop-Down Hotspots)

You can create sections of content that expand and collapse when you click a link or hot spot. This structure allows you to create items, such as bulleted lists, that are easy to scan, and then the users can expand individual items to get additional information. You can also use this structure to provide definitions. For more information about which output formats support this feature, see “Features Available in Each Output Format”.

When you identify the paragraph styles to start expand/collapse sections, you define whether those sections should initially be expanded (shown) or collapsed (hidden). ePublisher inserts an image indicating the state of the link. When a user clicks on a hotspot with the initial content collapsed, the content expands under the hotspot. If the user clicks a second time on the hotspot, the content is hidden again.

Using Styles and Markers for Expand/Collapse Sections

To use expand/collapse sections, you need the paragraph styles you want to start these sections, and you need the DropDownEnd marker in your source documents. Decide which paragraph styles should provide the link and start an expand/collapse section. For example, you could make Heading 4, Procedure Title, and Bullet Expand paragraph styles start expand/collapse sections.

To enable expand/collapse sections in your Stationery

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Paragraph Styles**, select the paragraph style you want to define as the link for the start of an expand/collapse section.
5. On the **Options** tab, set **Dropdown** to **Start open** or **Start closed**, based on whether you want the expand/collapse section to be displayed or hidden by default.
6. Repeat steps 3-4 for each paragraph style you want to start expand/collapse sections.



When writers use these styles, they can identify the end of each expand/collapse section with the DropDownEnd marker. All other paragraph styles should have the Dropdown option set to Continue to be included in the expand/collapse sections as needed.

Modifying Images for Expand/Collapse Sections

You can replace the default images ePublisher uses to indicate the state of an expanded or collapsed hotspot. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.

To modify expand/collapse images

1. *If you want to override the images for an output format*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Format Override Directory**.
 - b. Create the `Formats\formattype\Files\images` folder in your project folder, where *formattype* is the name of the output format to override, such as `Dynamic HTML`.
2. *If you want to override the images for a target*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Target Override Directory**.
 - b. Create the `Targets\targetname\Files\images` folder in your project folder, where *targetname* is the name of the target you want to override.
3. Paste the `.gif` files you want to use with names identical to those you want to replace in the `Images` folder you created. The following table lists the default images and their file names and sizes.

Image	Image Name	Image Size
	expanded.gif	Width: 8 pixels (0.111 inch) Height: 6 pixels (0.083 inch)
	collapse.gif	Width: 6 pixels (0.083 inch) Height: 8 pixels (0.111 inch)

Note: If you are generating with WebWorks Reverb output, you will modify the expand/collapse images in the `skin.png` file located in `Pages\images` in the Format or Target override directory for this output. For more information, See “Changing the Appearance of WebWorks Reverb”.

4. Regenerate your project to review the changes.

Defining Popup Windows

You can use popup windows to display brief additional information about a word or phrase, such as a glossary definition for a term in a topic. A popup window displays a link in a topic, which indicates to users they can hover over or click on the link, which displays the additional content. Popup windows streamline the initial content and allow users to choose whether they want to view the additional content.

To create a popup window, writers first create a link in the original text to the content it should display. Writers create this link using the link features in their source document application. Then, the writers can implement the popup window using markers or paragraph styles.

You need to assign popup behavior options to your paragraph and marker styles in your Stationery. For more information about which output formats support this feature, see “Features Available in Each Output Format”.

Using Marker Styles to Create Popup Windows

The default marker styles that define popup windows are `Popup`, `PopupOnly`, and `PopupEnd`. The writer inserts these markers into the source documents to specify what content to display through the popup window, and whether the popup content is displayed in only a popup window or in both a popup window and a clickable link that displays the content in a different topic. Some output formats support displaying the content only in a popup window.

Using Paragraph Styles To Create Popup Windows

You can define paragraph styles in your source documents to create popup windows. This approach avoids the need for markers, but you may need to create more styles in your source document templates. You can define both marker styles and paragraph styles to create different popup windows in your content.

To define paragraph styles for popup windows

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Paragraph Styles**, select the paragraph style you want to define as the first paragraph of a popup window.

This paragraph style is applied only to the first paragraph of content that should be displayed in a popup window. If a popup window may contain more than one paragraph of content, you need to create a second paragraph style and apply it to all paragraphs following the first paragraph that should be displayed in the popup window.

5. On the **Options** tab, select the appropriate value for the **Popup** option, such as **Define** or **Define with no output**. For more information about this option, click **Help**.
6. Select the paragraph style you want to define as one of the paragraphs of a popup window that follows the first paragraph of the popup window.
7. On the **Options** tab, select the appropriate value for the **Popup** option, such as **Append** or **Append with no output**. For more information about this option, click **Help**.

Assigning a Page Style to Popup Windows

If you have popup windows in your content, the appearance of the popup windows matches the rest of your topic pages, including breadcrumbs and company information. If you want to assign a different page style to your popup windows, you need to create a new page style and assign it to your popup paragraph styles.

Note: If you are using marker styles to create popup windows, you cannot use page styles to control the appearance of your popup windows. This process applies only to popup windows created with paragraph styles.

To assign a page style to popup windows

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. Create a new page style for popup windows by completing the following steps:
 - a. In **Page Styles**, click the **New Style** button. ePublisher adds a new page style called **Untitled**.
 - b. Enter the name for the new page style, such as `Popup Windows`.
 - c. On the **Properties** tab and the **Options** tab, select the options you want to assign to this page style.
5. In **Paragraph Styles**, select your popup paragraph style.
6. On the **Options** tab, set **Popup page style** to the page style you created for popup windows, such as `Popup Windows`.

Defining Related Topics

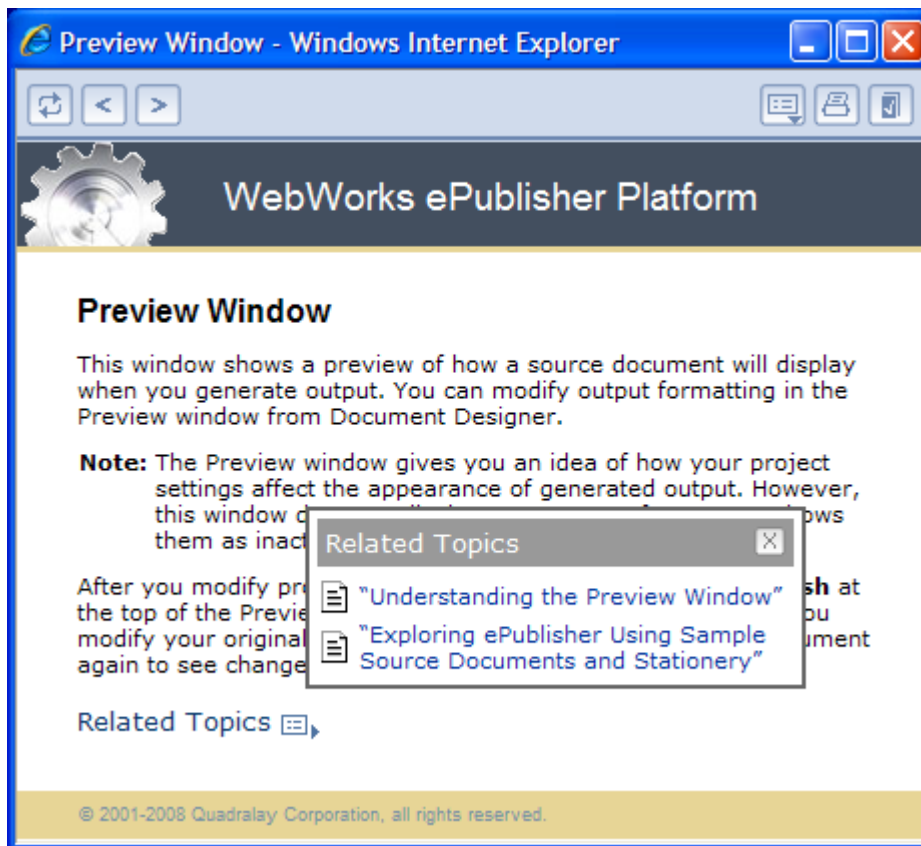
Related topics provide a list of other topics that may be of interest to the user of the current topic. For example, you could have a section called Creating Web Pages in your help. You may also have many other topics, such as HTML Tags and Cascading Style Sheets, that relate to creating Web pages. Identifying these related topics for users can help them find the information they need and identify additional topics to consider. However, providing these types of links as cross references within the content itself may not be the most efficient way to present the information. For more information about which output formats support this feature, see “Features Available in Each Output Format”.

Related topics and See Also links provide similar capabilities, but there are several important differences:

- Related topics can link to headings in a help set that do not start a new page.
- Related topics links are static and defined in the source documents as links. You must have all the source documents to create the link and generate the output.
- If a related topics list contains a broken link in the source document, that link is broken in the generated output. In a See Also link list, the broken link is not included in the output.

You can configure related topics to be displayed in the following ways:

- Included as a list in the topic itself.
- Displayed in a popup window when the user clicks a button, as shown in the following figure.



Note: If a related topic link is broken in the source document, in most cases that link is broken in the generated output. WebWorks Help and WebWorks Reverb provide an additional feature by removing broken links from related topics lists that are displayed in a popup window when a user clicks the Related Topics button.

Using a Paragraph Style for Related Topics Lists

You can use a paragraph style to define a related topics list. Create a unique paragraph style to use specifically for the related topics list items. The writer should create a list of links in a topic in the source document, and apply the paragraph style to each list item.

To define a paragraph style for a related topics list

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. In **Paragraph Styles**, select the paragraph style you want to define as the related topics list. This paragraph style is applied to all paragraphs in a related topics list.
5. On the **Options** tab, select the appropriate value for the **Related topic** option. For more information about this option, click **Help**.
 - To display the list of related topics in the body of the topic, select **Define**.
 - To display the list of related topics only when the **Related Topics** button is clicked, select **Define with no output**.

Note: The **Show related topics inline button** and the **Show related topics toolbar button** target settings specify whether to include related topics buttons and where to include them. If you select **Define** for your related topics paragraph style and you enable the **Show related topics inline button** setting, both the list of related topics and the related topics button itself are displayed in the topics with related topics.

Defining See Also Links

See Also links are associative links (Alinks) that identify other topics that may be of interest to the user of the current topic. These links use internal identifiers to define the links and the list is built dynamically based on the topics available when the user displays the links. See Also links are important to use with larger help sets and merged help sets. For more information about which output formats support this feature, see “Features Available in Each Output Format”.

Related topics and See Also links provide similar capabilities, but there are several important differences:

- See Also links must link to styles that start a new topic, such as a heading.
- See Also links are dynamic and the lists of links are built at display time instead of during help generation.
- Since See Also link lists are dynamically built, they do not include links to topics that are not available when the user displays the links. If a related topics list contains a broken link in the source document, that link is broken in the generated output for most output formats.

Enabling See Also Functionality

If you want to create a See Also link as inline text without a button, create a unique character style and select the **See Also** option for that style. If you want to use a button to display the See Also links, create a unique paragraph style, select the **See Also** option for that style, and type the See Also text on that paragraph. The properties you select for the paragraph style in Style Designer affect the text of the **See Also** button. To modify the appearance of the button itself, see “Modifying the Appearance of the See Also Button”. You also need to create a SeeAlsoLink and SeeAlsoKeyword marker.

HTML Help also supports the SeeAlsoLinkDisplayType and SeeAlsoLinkWindowType markers. These markers allow you to change how the See Also links are displayed in HTML Help. For example, you can display the links as a popup menu.

Note: If paragraph and marker styles are created in your source document after you create a project, scan the document in the project again for the changes to take effect.

To enable See Also functionality in your Stationery

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is an output format that supports this option.
4. *If you want to use buttons for See Also links*, complete the following steps:
 - a. In **Paragraph Styles**, select the paragraph style to use for See Also links.
 - b. On the **Options** tab, set **See Also** to **Enabled**.
5. *If you want to use inline text for See Also links*, complete the following steps:
 - a. In **Character Styles**, select the character style to use for See Also links.
 - b. On the **Options** tab, set **See Also** to **Enabled**.
6. In **Marker Styles**, select the SeeAlsoKeyword marker style.
7. On the **Options** tab, set **Marker type** to **See Also Keywords**.
8. In **Marker Styles**, select the SeeAlsoLink marker style.
9. On the **Options** tab, set **Marker type** to **See Also Link Keywords**.

Modifying the Appearance of the See Also Button

The properties you select for the paragraph style in Style Designer affect the text of the **See Also** button. In some output formats, you can modify the color of the See Also button background and borders. You must separately modify each button border. You can also modify each of the See Also button colors.

Note: To change the color of the See Also button, you modify the `content.xml` file. *If you modify the `content.xml` file*, you will be responsible for maintaining your customizations to the file as needed each time you update your Stationery to work with a new version of ePublisher.

For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.

To change the color of the See Also button

1. *If you want to override the processing for an output format*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Format Override Directory**.
 - b. Create the `Formats\formattype\Transforms` folder in your project folder, where *formattype* is the name of the output format you want to override, such as `WebWorks Help 5.0`.
2. *If you want to override the processing for a target*, complete the following steps:
 - a. In your Stationery design project, on the **View** menu, click **Target Override Directory**.
 - b. Create the `Targets\targetname\Transforms` folder in your project folder, where *targetname* is the name of the target you want to override.
3. Copy the `content.xml` file from the following folder to the override folder you created within your project folder:

```
Program Files\WebWorks\ePublisher Designer\Formats\formattype\Transforms
```

```
<html:table border="0" cellspacing="0" cellpadding="0" onclick="{ $VarCargo/
wwalinks:ALink[1]/@onClick}" summary="">
  <html:tr>
    <html:td height="2" colspan="4" bgcolor="#FFFFFF"></html:td>
    <html:td width="2" height="2" background="{ $Var_seertup}"></html:td>
  </html:tr>
  <html:tr>
    <html:td width="2" height="2" bgcolor="#FFFFFF"></html:td>
    <html:td height="2" colspan="3" bgcolor="#EEEEEE"></html:td>
    <html:td width="2" height="2" background="{ $Var_seeright}"></html:td>
  </html:tr>
  ...

```

6. Modify the RGB color values in the `bgcolor` attributes within this table to adjust the colors of the margins that form parts of the **See Also** button.
7. Save the `content.xml` file.
8. Regenerate your project to review the changes.

Define the Default Settings for Each Target

You can have one or more output formats in your Stationery. You can also define multiple targets in your Stationery. The Stationery Designer properties and options are shared across all targets and output formats. Some settings, such as the target settings, variable values, conditions, and cross-reference formats, are defined per target. Some targets and output formats also offer additional features and customizations.

For each target in your Stationery, define the following default settings in your Stationery. If a writer installs the support with ePublisher Express, that writer can modify these settings in each project based on the Stationery:

- Target settings, such as the company information and navigation on each page. To specify the target settings for a target, select the target, and then click **Target Settings** on the **Target** menu. For more information about a setting, click **Help**.
- Index settings. For more information, see “Defining the Default Index Settings”.
- Variable values. For more information, see “Defining the Default Processing of Variables”.
- Condition visibility. For more information, see “Defining the Default Processing of Conditions”.
- Cross Reference formats and rules. For more information, see “Defining the Default Processing of Cross References”.
- PDF integration. For more information, see “Defining Default PDF Generation Settings”.
- Output format-specific customizations and features. For more information, see “Customizing Stationery for Output Format-Specific Features”.
- Reporting options, such as accessibility reporting. For more information, see “Defining the Accessibility Report to Validate Content” and “Defining Other Reporting Options”.

Defining the Default Index Settings

The index provides the user with a point-and-click resource for quickly navigating your online content. ePublisher generates the index by default for the available formats, using the native indexing features of the source document tool used to create the printed index.

The groups and order of index entries in your online index are determined through index entries defined in the source document, the `locales.xml` file in the ePublisher installation folder, and the output format display environment. The `locales.xml` file also defines the text that identifies `See` and `See also` style entries in your index. For more information about customizing the index appearance, see “Modifying the Appearance of the Index in Dynamic HTML”.

Depending on your output format, you can specify the file name for the generated index whether to generate the index. With the power of many full-text search engines, you may choose not to include an index in your generated output.

Note: If you selected the WebWorks Help, WebWorks Reverb, or WinHelp output format, you must generate an index with the given file name. These options are predefined and cannot be changed.

To enable index generation in your online content

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. Set **Generate index** to **Enabled**.
4. *If you want to change your index file name and your output format supports it*, specify the new file name in **Index filename**.

Defining the Default Processing of Variables

A variable provides a placeholder for standard terms and for names that may change. By defining variables in your source documents, you have global control over the values contained within those variables. For example, you can create a variable for the copyright date of your documentation. You can then use that variable as needed in your content. Each year when you need to update the copyright, you can change the variable value in a single location instead of using the search and replace method through your documents.

In your project, you can specify the value of any variable. When you change the value of a variable in your project, it changes the value only in your generated output. The variable value is not changed in your source documents. You can also use the value of the variable defined in your source documents. To use the value defined in a source document for a variable, select **Use Document Value** for that variable.

To specify a variable value

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Variables**.
3. Find the variable you want to modify.
4. In the **Value** column, select its current definition and make the desired changes, or select **Use Document Value**.

Defining the Default Processing of Conditions

In your source documents, you can define conditions and use them to show or hide parts of your content. ePublisher allows you to define the visibility for each condition in your project. The conditions that are available in your project come directly from your source documents. You can modify the value for any of your conditions, which affects how your conditional text is incorporated into your generated output. You can also select **passthrough** for a condition to insert the content directly into your output without being transformed and coded for your output format. This option allows you to directly add HTML coding or multimedia files to your output.

Note: You can also use PassThrough markers and the Pass Through paragraph style and character style options to insert content directionly into your output without being transformed and coded for your output.

To modify the value of a condition

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Conditions**.
3. Find the condition you want to modify.
4. In the **Value** column, select the visibility option you want to use for that condition.

Defining the Default Processing of Cross References

Cross-references help users navigate through your content. ePublisher automatically transforms cross-references to links in the generated output. However, you often want cross-references in your online content to use a different format than your printed content. For example, you usually include page numbers only in your printed content. ePublisher enables you to add, edit, and delete cross-reference formats for your online output.

Note: This option is available only for specific source document types and each defined cross reference rule applies only to the selected source document type.

To define and manage cross-reference formats

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Cross Reference Rules**.
3. In **Document type**, select the source document type for which you want to define the cross-references.
4. *If you want to add a cross reference format*, complete the following steps:
 - a. Click **Add New Cross Reference**.
 - b. In **Name**, type the format or name of the cross reference format you want to define. For example, to remove the phrase `see page` followed by a page number from a Word source document, type: `see page {PAGEREF \h}`. You can find the correct syntax for the search pattern by inspecting your source documents for the values.
 - c. In **Replacement**, type the format or text you want to replace the format you specified in **Name**. To replace the format with nothing, leave the **Replacement** field blank.
 - d. Click **OK**.
5. *If you want to modify a defined cross-reference format*, complete the following steps:
 - a. Double-click an existing cross-reference format you want to modify.
 - b. In **Replacement**, type the format or text you want to replace the format you specified in **Name**. To replace the format with nothing, leave the **Replacement** field blank.
 - c. Click **OK**.
6. *If you want to change the order in which ePublisher processes the cross-reference formats*, select a cross reference rule in the list, and then use the arrow buttons to arrange the formats from top to bottom in the order you want ePublisher to process them:
7. *If you want to delete a defined cross-reference format*, select the cross reference rule in the list, and then click **Delete Cross Reference**:
8. Click **OK** to close the Cross Reference Rules window.

Defining Default PDF Generation Settings

ePublisher provides PDF options that enable you to generate a PDF file for each source document or to generate a PDF file for each top-level group in a project. If you generate a PDF for each top-level group, ePublisher combines all the source documents within a single top-level group, and then generates a single PDF file for those documents.

You can deliver the PDF files separate from your help system, or in some output formats you can add a PDF button to your toolbar that displays the PDF files in the window. The PDF file displayed when the user clicks the PDF button depends on which PDF generation options you chose in your project. If you have selected to:

- Generate PDF files only for the top-level groups, the window displays the PDF file for the top-level group in which the currently viewed topic is located when the user clicks the PDF button.
- Generate PDF files only for each source document, the window displays the PDF file for the source document in which the currently viewed topic is located when the user clicks the PDF button.
- Generate PDF files for all source documents and for all top-level groups, the window displays the PDF file for the source document in which the currently viewed topic is located when the user clicks the PDF button.

To specify the PDF file generation settings and enable the PDF toolbar button

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. *If you want to generate a PDF file for each source document*, set **Generate a PDF per document** to **Enabled**.
4. *If you want to generate a PDF file for each top-level group of source documents*, set **Generate a PDF per top level group** to **Enabled**.
5. Set **Show PDF button** to **Enabled**.
6. Click **OK**.

Defining the Accessibility Report to Validate Content

Content that must be accessed by people with disabilities must conform to guidelines published by both the W3C and the United States government. These guidelines are intended to help authors produce accessible content. ePublisher helps you produce online content that conforms to the W3C Web Content Accessibility Guidelines 1.0 (WCAG), Section 508 of the U.S. Rehabilitation Act of 1998, and the Americans with Disabilities Act (ADA).

If you take certain steps in creating your source documents and setting up your project, your generated output is accessible through assistive technologies such as screen readers. When you generate your project, ePublisher can perform several checks to verify that you embedded information and conform to the accessibility standard in these areas:

- Alternate text for all images (ImageAltText marker)
- Alternate text for clickable regions in all image maps (ImageAreaAltText marker)
- Long descriptions for all images (ImageLongDescText and ImageLongDescByRef markers)
- Summaries for all tables (TableSummary marker)

Note: ePublisher does not check to ensure that you have provided expansion text for abbreviations or acronyms nor does it verify that you have included citation markers for quotations. You can use the AbbreviationTitle, AcronymTitle, and Citation markers to add this information to your content.

For more information about producing accessible content, and to check your content further for compliance, see the following Web sites:

- www.w3c.org/TR/WCAG10-CORE-TECHS
- www.w3.org/WAI
- www.w3.org/WAI/Policy/

ePublisher does not perform accessibility validation by default. You must enable accessibility validation of the content. ePublisher validates that all images and image maps have alternate text, all images have long descriptions, and all tables have summaries. You can choose which accessibility validation checks you want ePublisher to run.

Note: If you disable any of the accessibility validation checks, you cannot consider your content to be accessible or Section 508 compliant.

To define the accessibility report that validates online content for accessibility compliance

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. Set **Generate accessibility report** to **Enabled**.
4. *If you want to exclude the alternate text check for images*, disable **Validate accessibility image alternate tags**.
5. *If you want to exclude the alternate text check for image maps*, disable **Validate accessibility image map alternate tags**.
6. *If you want to exclude the long description check for images*, disable **Validate accessibility image long descriptions**.
7. *If you want to exclude the summary check for tables*, disable **Validate accessibility table summaries**.

When a project based on this Stationery is generated, the accessibility report is created with the information you selected to include in that report.

Defining Other Reporting Options

In addition to the accessibility reporting features, ePublisher provides reports to help you identify and resolve potential transformation issues. You can define which conditions are informational messages, warnings, or errors. Errors force ePublisher AutoMap to return a non-zero return code and stops the output deployment process.

To define the other report options

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. Review and define the settings for the following categories of report settings:
 - **Filenames Report**
 - **Links Report**
 - **Styles Report**
 - **Topics Report**
4. Click **OK**.

Saving and Testing Stationery

Once you have defined your Stationery design project, you need to save and test the Stationery. This process allows you to adjust the Stationery as needed before further customizing your design and deploying the Stationery for use.

Note: Document-specific information, such as groups, documents, and changes made with Document Designer, are not saved in Stationery files.

When you save the Stationery design project as Stationery, ePublisher stores all the style information, settings, and definitions from the project in the Stationery. ePublisher also copies the user, output format, and format target override files, and saves them as part of your Stationery.

To save your Stationery design project as Stationery and test it

1. Open your Stationery design project.
2. On the **File** menu, click **Save as Stationery**.
3. Specify the Stationery name, location to store the Stationery, and the targets to include in the Stationery. For more information about an option, click **Help**.
4. Click **OK**.
5. Close ePublisher Designer.
6. Open ePublisher Express.
7. On the **File** menu, click **New Project**.
8. Specify the project name and the location to store the project.
9. In **Stationery**, specify your Stationery file, and then click **Next**.
10. Add your standard sample source document to the project, and then click **Finish**. For more information about an option, click **Help**.
11. On the **Project** menu, click **Scan All Documents**.
12. On the **Project** menu, select the active target you want to test.
13. On the **Project** menu, click **Generate All**.
14. Review the generated output.
15. Repeat steps 12-14 for each target in your project.

Once you finish customizing your Stationery, store it in a central location where writers can use it. If you move the Stationery, ePublisher Express notifies the writers when they open their projects that it cannot find the Stationery associated with the project. The writers then need to update their projects to use the Stationery in its new location.

Customizing Your Design

By defining online navigation, setting options and preferences in your project, and embedding additional information in your source documents, you can define the behavior and appearance of your online content. The following sections provide some additional information about techniques you can use to further refine and customize your design.

Note: These techniques are supported in all formats unless otherwise noted.

Understanding Customized Processing

ePublisher uses file and folder locations to provide a structure where you can create and store override files. These files customize how ePublisher transforms your content.

When you generate output for an output format, ePublisher uses the following process to identify the files to use to process your content and complete the task:

1. ePublisher checks the `Targets` folder hierarchy in your project for the files required to complete the task. ePublisher checks the folder hierarchy named for the target you are generating.
2. If the files are not found in the `Targets` folder hierarchy in your project, ePublisher checks the `Formats` folder hierarchy in your project for the files required to complete the task. ePublisher checks the folder hierarchy named for the output format of the target you are generating.
3. If the files are not found in your project folder hierarchy, ePublisher checks the installation folders.

This process allows you to override any default file in the installation folder hierarchy for one or more output formats or targets by placing a customized file with the same name at the correct location in the project folder hierarchy.

By recreating the file path in the `Targets` folder in your project, and storing a modified file in the correct location, you can change the processing for that specific target. This method allows you to customize a specific target in a project that has multiple targets using the same output format.

By recreating the file path in the `Formats` folder in your project, and storing a modified file in the correct location, you can change the processing for all targets that use the same output format. This method is also helpful for projects with one target.

Note: Do not modify files in the installation folders. Store customized files only in the project folder hierarchy. You need to create the `Targets` and `Formats` folder hierarchies in your project folder, as needed for the files you want to override.

For more information, see “Understanding Stationery, Projects, and Overrides” and “Understanding How ePublisher Works”.

Disabling External CSS Files

By default, ePublisher creates an external CSS file for each source document in your project. Depending on how many source documents are in the project and the amount of style information in each document, these CSS files can become large and redundant. ePublisher allows you to use a custom CSS file to avoid this issue. ePublisher still creates these external CSS files, but you can simply delete the unused CSS files. However, if you are deploying your generated content directly to a Web server, you may not want the unused CSS files deployed with your content.

You can override the `pages.fti` and `styles.xml` files and modify them to no longer create the external CSS files.

Note: *If you override the `pages.fti` and `styles.xml` files and modify them to no longer create the external CSS files*, you will be responsible for maintaining your customizations to these files as needed each time you update your Stationery to work with a new version of ePublisher.

To disable external CSS files

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. *If you want to override the processing for an output format*, create the `Formats\formattype\Transforms` folder in your project folder, where `formattype` is the name of the output format you want to override, such as `Dynamic HTML`.
3. *If you want to override the processing for a target*, create the `Targets\targetname\Transforms` folder in your project folder, where `targetname` is the name of the target you want to override.
4. Copy the `pages.fti` and the `styles.xml` files from the following folder to the override folder you created within your project folder:

```
Program Files\WebWorks\ePublisher Designer\Formats\formattype\Transforms
```

5. Open the `pages.fti` file you copied to your project override folder in a text editor.
6. Under the `<Groups>` tag, add the following code:

```
<Group name="css" />
```

7. Under the `<Settings>` tag, add the following code:

```
:<!-- Emit -->
<!-- -->
<Setting name="emit-external-css" group="css" default="true">
<SettingClass name="boolean" />
</Setting>
```

8. Save and close the `pages.fti` file.
9. Open the `styles.xml` file you copied to your project override folder in a text editor.
10. Under the `<wwfiles:Files version="1.0">` tag, add the following code:

```
:<xsl:if test="wwproject:GetFormatSetting('emit-external- css', 'false') = 'true'">
```

11. Under the `<!-- Call template -->` tag, replace the following line:

```
<xsl:variable name="VarResult">
```

with the following line:

```
<xsl:variable name="VarResultAsXML">
```

12. Under the `<!-- Call template -->` tag, and just before the `<xsl:value-of select="wwexsldoc:Document($VarResult,$VarPath,'utf-8','text')"/>` tag, add the following code:

```
<xsl:variable name="VarResult" select="msxsl:node-set($VarResultAsXML)"/>
```

13. Under the `<xsl:variable name="VarProgressProjectGroupsEnd" select="wwprogress:End()"/>` tag, add the following code:

```
</xsl:if>
```

14. Save and close the `styles.xml` file.
15. On the **Project** menu, select the **Active Target** you want to specify settings for.
16. On the **Target** menu, click **Target Settings**.
17. Set **emit-external-css** to **Disabled**.

Mark of the Web

When a user reads a locally installed help system displayed through the Microsoft Internet Explorer (MSIE) browser, the browser can apply local machine zone security to the help system content pages if they are loaded directly from a file system as opposed to a web server. This security can interfere with some features, such as those implemented using JavaScript, and display security windows. For example, the user may see Javascript security windows when using MSIE to directly open pages generated from either of the WebWorks Help, WebWorks Reverb, or Dynamic HTML formats.

The Mark of the Web (MOTW) is a technique to avoid these security windows generated by MSIE. However, while the MOTW suppresses security windows, it also disables links to baggage files on the local file system such as PDF and image files, so this is something to consider when making use of this function within WebWorks Help 5.0, WebWorks Reverb, or Dynamic HTML. For more information about the Mark of the Web, see msdn.microsoft.com.

Note: Due to the continued evolution of the Microsoft Windows platform along with subsequent releases of MSIE, if you are enabling MOTW for your generated output, you will need to pay particular attention to the version of ePublisher formats that you are using for your conversions. For example, with the release of Microsoft Windows 8, a modification to the MOTW is required in order to load non-HTML files. This modification has been included in all ePublisher formats beginning with format versions 2013.2. So if you are using a prior version of the format, you should consider upgrading your format to version 2013.2 or later.

To enable the Mark of the Web for WebWorks Reverb or WebWorks Help

1. On the **Target** menu, click **Target Settings**.
2. Expand **File Processing**.
3. Set **Insert Mark of the Web (MOTW)** to **Enabled**.

To enable the Mark of the Web for other HTML based formats

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. *If you want to override the processing for an output format*, create the `Formats\formattype\Pages` folder in your project folder, where *formattype* is the name of the output format you want to override, such as `Dynamic HTML`.
3. *If you want to override the processing for a target*, create the `Targets\targetname\Pages` folder in your project folder, where *targetname* is the name of the target you want to override.
4. Copy the `Page.asp` file from the following folder to the override folder you created within your project folder:

Program Files\WebWorks\ePublisher Designer\Formats\formattype\Pages

5. Open the `Page.asp` file you copied to your project override folder in a text editor.
6. Find the following line of code:

```
<!--MOTW-DISABLED saved from url=(0014)about:internet -->
```
7. Delete `MOTW-DISABLED` from this line of code.
8. Save and close the `Page.asp` file.

9. *If you want to override the processing for WebWorks Help*, complete the following steps:
 - a. Search each HTML file in the WebWorks Help support files for `MOTW-DISABLED`.
 - b. Override each of these files in your project with a copy of the file with `MOTW-DISABLED` removed from the file.
10. Regenerate your project and verify the results.

Customizing the Content Page Design

You can customize the appearance of your content page to meet your specific needs. ePublisher provides several customization options, such as where to display company information, browse buttons, and breadcrumbs. These options allow you to achieve a professional result in your generated output.

You may want to further customize your content page design. ePublisher provides this flexibility by allowing you to override the `Page.asp` file, which defines the appearance and behavior of your content page. However, if you customize this file, some customization options in the ePublisher console may no longer affect your content page design.

For example, you can override the `Page.asp` file to add a graphic bar across the top of the page with the product logo and name. You can also add a graphic bar across the bottom of the page with a copyright and a link to your company Web site and customer support area. You can also modify the formatting and layout of the default page design options, such as the company information options.

To override the `Page.asp` file

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. *If you want to override the processing for an output format*, create the `Formats\formattype\Pages` folder in your project folder, where *formattype* is the name of the output format you want to override, such as `Dynamic HTML`.
3. *If you want to override the processing for a target*, create the `Targets\targetname\Pages` folder in your project folder, where *targetname* is the name of the target you want to override.
4. Copy the `Page.asp` file from the following folder to the override folder you created within your project folder:

```
Program Files\WebWorks\ePublisher Designer\Formats\formattype\Pages
```
5. Open the `Page.asp` file you copied to your project override folder in a text editor.
6. Modify the `Page.asp` file as needed.
7. Save and close the `Page.asp` file.

Customizing Output Format-Specific Features

Some output formats provide additional features and ways you can customize those formats for your specific needs. For example, you can customize the splash page, search processing, and window text in WebWorks Help output. You can also customize overall window design in WebWorks Help and HTML Help. For more information, see “Customizing Stationery for Output Format-Specific Features”.

Backing Up Your Stationery Design Project, Stationery, and Projects

You should save a backup copy of your Stationery design project, your Stationery, and any individual projects you have. These snapshots can help you revert your project to a specific stage at a point in time. This snapshot can also help you if your project or Stationery become corrupted or lost.

Element to Back Up	Description of Process and Considerations
Stationery design project	Copy all your sample files, the <code>.wep</code> file, and the <code>Files</code> , <code>Formats</code> , and <code>Targets</code> folders and subfolders. Maintain the same structure of these files and folders.
Stationery	You can create a <code>.zip</code> file of your Stationery to back it up. If needed, you can also recreate your Stationery from your Stationery design project. Copy the <code>.wxsp</code> file, the <code>.manifest</code> file, and the <code>Files</code> , <code>Formats</code> , and <code>Targets</code> folders and subfolders. Maintain the same structure of these files and folders.
Individual project	To recreate your generated output, you need the source documents, the Stationery you used, and your individual project file. Copy your source documents and the <code>.wrp</code> file. Maintain the same structure of your source files and the <code>.wrp</code> file.

Deploying Stationery

Once you have saved, tested, and finished customizing your Stationery, you need to deploy it so writers can use the Stationery for their projects. Review the following considerations when deploying your Stationery for writers to use in their projects:

- Store your Stationery in a central location where writers have read access.
- When you save the Stationery design project as Stationery, ePublisher stores all the style information, settings, and definitions from the project in the Stationery. ePublisher also copies the user, output format, and format target override files, and saves them as part of your Stationery.
- If you move the Stationery, ePublisher Express notifies the writers when they open their projects that it cannot find the Stationery associated with the project. The writers then need to update their projects to use the Stationery in its new location.
- Make sure your source document templates and standards support your Stationery. Keep these files updated as a unit. If you add a style to your Word or FrameMaker template, also add it to your Stationery design project and update your Stationery.
- If the source documents for an existing project use only the styles in your standard templates, writers can use ePublisher Express to synchronize their project with your Stationery. This process allows you to update the project using the current Stationery for the project, or you can select a different Stationery to associate with the project.
- Schedule a training session with the team about each of the features they can use in their help, how to use styles and markers to define those features, and how to use ePublisher Express and their projects to achieve the results they need.
- Make sure your Stationery has the output formats and targets you need. Since all output formats in the Stationery share Style Designer properties and options, if you want different output formats to have different behaviors or appearance, you may need to create and maintain more than one Stationery.
- Determine whether writers should change target-specific settings, such as variables, conditions, cross reference rules, and target settings in their projects. When writers install ePublisher Express, they can specify whether to enable these menu options.

To save your Stationery design project as Stationery

1. Open your Stationery design project.
2. On the **File** menu, click **Save as Stationery**.
3. Specify the Stationery name, location to store the Stationery, and the targets to include in the Stationery. For more information about an option, click **Help**.
4. Click **OK**.

Managing and Updating Stationery

Once you deploy your Stationery, manage its use to make sure it continues to meet your needs. Your source document templates and standards can change over time. Make sure your source document templates support your Stationery and keep these files updated as a unit. If you add a style to your Word or FrameMaker template, also add it to your Stationery design project and update your Stationery.

You may also decide to add a feature to your output, such as expand/collapse sections or popup windows. To add a feature, you may need to make changes to both your source document templates and your Stationery. Then, you need to put together a deployment or roll-out plan to help writers decide when and how each project should use the features.

If you move or change the Stationery, ePublisher Express notifies the writers when they open their projects. The writers have the opportunity to synchronize their projects with the Stationery and bring their projects inline with your new standards. For more information, see “Synchronizing Projects with Stationery”.

Be careful when you update your Stationery to make sure you have the files you need. Review the following considerations for properly maintaining your Stationery:

- Store your Stationery design project and supporting files in a version control system. This process allows you to monitor how it changes over time and ensures you can return to a previous version, if needed.
- Create a subfolder in your Stationery design project and store a sample of each source document type in that folder. These files help you test and verify the output as you modify your source document styles and your ePublisher styles and settings.
- Do not directly open and modify the Stationery files. To make sure your Stationery is properly updated, always open the Stationery design project for the Stationery, make your changes, and then save a new copy of the Stationery.
- Consider saving your updated project as Stationery to a new location and have several writers test some smaller projects with the updated Stationery before you update your Stationery for all projects.

When you save your Stationery, ePublisher creates the following folders that contain information about any customizations or overrides you created when you developed the Stationery:

- *StationeryName*\Formats\OutputFormat
- *StationeryName*\Formats\OutputFormat.base

where *StationeryName* is the name you specified for the Stationery, and *OutputFormat* is the type of output format you specified for a target in the Stationery. You can use these folders to help you identify any customizations or overrides you specified for your Stationery when updating your Stationery.

The *StationeryName*\Formats\OutputFormat folder contains any customizations or overrides you specified when designing the Stationery. ePublisher Express synchronizes with the files in the *OutputFormat* folder and uses the information about customizations and overrides contained in files in the *OutputFormat* folder to generate output.

Note: The Stationery may have one or more *OutputFormat* folders, based on the settings you specified in your Stationery.

The *StationeryName*\Formats\OutputFormat.base folder contains copies of all the files located in the \Program Files\WebWorks\ePublisher\2018.1\FFormats\OutputFormat folder. These files define the default output format and transforms and are installed by default when you install ePublisher.

You can do a compare, or **diff**, between the files located in these folders to quickly see any customizations or overrides specified for the Stationery. You can use this information to help you reapply customizations and overrides as needed when designing a newer version of the Stationery in ePublisher Designer.

To update your Stationery

1. Open your Stationery design project.
2. Make the desired changes.
3. On the **File** menu, click **Save as Stationery**.
4. Specify the Stationery name, location to store the Stationery, and the targets to include in the Stationery. To replace the existing Stationery, specify the same name and values as the existing Stationery. You do not have to replace the existing Stationery. You can also create a new Stationery with a different name, such as by adding a version number to the Stationery name.
5. Click **OK**.

6

Customizing Stationery for Output Format-Specific Features

Some output formats provide additional features and ways you can customize those formats for your specific needs. For example, you can customize the splash page, search processing, and window text in WebWorks Help output. You can also customize overall window design in WebWorks Help, WebWorks Reverb, and Microsoft HTML Help. The following sections summarize these customizations and methods.

Customizations Specific to WebWorks Help

You can customize the appearance and behavior of WebWorks Help in several ways. For example, you can customize the colors and images, and toolbar buttons displayed in the browser. You can also define which buttons are included in the toolbar pane. The following sections describe these WebWorks Help-specific customizations. For more information about changes you can make to multiple output formats, see “Checklist: Design, Deploy, and Manage Stationery”.

Renaming the Top-Level Entry File

When you generate output for your project, a top-level entry file is created. This file defines the frameset for the help and gives the user a file to open that displays the complete help. By default, the top-level entry file is named `index.html`, but you can specify any name you need for the top-level entry file as long as you specify an `.html` or `.htm` extension.

To rename the top-level file

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. In **Top level filename**, type the name you want to use for the top-level entry file.
4. Click **OK**.
5. Regenerate the project and review the results.

Selecting a Theme

In WebWorks Help, you can select a theme (skin) to define the appearance of your help. ePublisher provides many different WebWorks Help themes. Each theme provides a custom skin, which defines the toolbar color, button images, navigation pane, tabs, and splash image.

To choose a theme for WebWorks Help

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. Click on the **WebWorks Help** setting to reveal its contents.
4. In **Theme**, select the appropriate value.
5. Click **OK**.
6. Regenerate the project and review the results.

You can further customize your theme to deliver the specific appearance you want. For example, you can override images in your selected theme.

Customizing the Splash Page in WebWorks Help

You can replace or remove the splash page that is displayed while your WebWorks Help opens. You can customize the splash page in the Stationery, and then writers can override this customization in each project, as needed.

Replacing the Splash Image

The splash page is the first page that displays in the topic pane when the help set launches initially. By default, WebWorks Help systems display the WebWorks splash image. You can replace the default splash image with a custom image.

To replace the splash page image

1. Identify the theme in use for your WebWorks Help target that you want to modify. For more information, see “Selecting a Theme”.
2. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
3. *If you want to override the image for all WebWorks Help targets*, create the **Formats**\WebWorks Help 5.0\Pages\images folder in your *projectname* folder, where *projectname* is the name of your ePublisher project.
4. *If you want to override the image for one WebWorks Help target*, create the **Targets**\WebWorks Help 5.0\Pages\images folder in your *projectname* folder, where *projectname* is the name of your ePublisher project.
5. Copy the `splash.jpg` file from the following folder to the `images` override folder you created within your project folder:
`Program Files\WebWorks\ePublisher Designer\Formats\WebWorks Help 5.0\ Pages\images`
6. Open the `splash.jpg` file you copied to your project override folder and modify it to be the splash page image you want.
7. Save and close the `splash.jpg` file.
8. Regenerate your project to review the changes.

Removing the Splash Page

When WebWorks Help opens, it displays the splash page. However, instead of displaying the splash page, you can configure WebWorks Help to display the first topic in the help.

To remove the splash page

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. Set **Show first document instead of splash page** to **Enabled**.
4. Click **OK**.
5. Regenerate the project and review the results.

Customizing the Toolbar in WebWorks Help

You can customize the toolbar pane in WebWorks help to provide the buttons and options you need. You can add, remove, and replace toolbar buttons. You can also customize the appearance of the toolbar pane. For more information about the toolbar pane, see “Toolbar Pane in WebWorks Help”.

Adding and Removing Toolbar Buttons in WebWorks Help

You can add and remove toolbar buttons from the toolbar pane in WebWorks Help. To add a button, set the format setting for that button to **Enabled** for your WebWorks Help target. To remove a button, set the format setting for that button to **Disabled** for your WebWorks Help target.

To add or remove one or more toolbar buttons from the toolbar pane

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. Set the following target settings to **Enabled** or **Disabled** to define whether a button is displayed in the toolbar pane:

Button	Format Setting to Modify
Show in Contents	Automatically synchronize in TOC
Bookmark	Show bookmark toolbar button
PDF	Show PDF button
Previous and Next browse buttons	Show previous & next toolbar buttons
Print	Show print toolbar button
Related Topics	Show related topics toolbar button

4. Click **OK**.

Replacing the Toolbar Buttons in WebWorks Help

WebWorks Help provides several default buttons. These buttons allow the user to navigate through the help, and they provide additional features, such as printing a page from the help or emailing a link to a topic page. The toolbar buttons are .gif images for which you can provide override files. You can replace these standard buttons with other .gif images.

To replace one or more toolbar buttons in WebWorks Help

1. Identify the theme in use for your WebWorks Help target that you want to modify. For more information, see “Selecting a Theme”.
2. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
3. *If you want to override the images for all WebWorks Help targets*, create the WebWorks Help 5.0\Skins\theme\Files\wwhelp\wwhimpl\common\images folder in your *projectname***Formats** folder, where *theme* is the name of the WebWorks Help theme you want to override, such as Lobby_Blue, and *projectname* is the name of your ePublisher project. If the theme name is two words, include an underscore instead of a space between the words.
4. *If you want to override the images for one WebWorks Help target*, create the WebWorks Help 5.0\Skins\theme\Files\wwhelp\wwhimpl\common\images folder in your *projectname***Targets** folder, where *theme* is the name of the WebWorks Help theme you want to override, such as Lobby_Blue, and *projectname* is the name of your ePublisher project. If the theme name is two words, include an underscore instead of a space between the words.
5. Paste the .gif files you want to use with names identical to those you want to replace in the images folder you created. The following table lists several default button images and their file names. For a complete list of image file names, see the appropriate folder in the ePublisher installation folder within Program Files. You can copy the files from the installation folder and then modify them in your project.

Button	File Name	Description
Show in Contents	sync.gif	Highlights the currently displayed topic in the table of contents.
Show in Contents (not available)	syncx.gif	Displayed when a topic is not currently displayed in the topic pane.
Previous	prev.gif	Displays the previous topic in the help.
Previous (not available)	prevx.gif	Displayed when there is no previous HTML document available.
Next	next.gif	Displays the next topic in the help.
Next (not available)	nextx.gif	Displayed when there is no next HTML document available.

6. Regenerate your project to review the changes.

Changing the Background Color of the Toolbar

The toolbar in WebWorks Help is composed of a single repeating .gif image. This image is located in the images folder of your WebWorks Help theme. To modify the background color of the toolbar, you can override the toolsbg.gif file.

To change the background color of the toolbar

1. Identify the theme in use for your WebWorks Help target that you want to modify. For more information, see “Selecting a Theme”.
2. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
3. *If you want to override the image for all WebWorks Help targets*, create the WebWorks Help 5.0\Skins\theme\Files\wwhelp\wwhimpl\common\images folder in your *projectname***Formats** folder, where *theme* is the name of the WebWorks Help theme you want to override, such as Lobby_Blue, and *projectname* is the name of your ePublisher project. If the theme name is two words, include an underscore instead of a space between the words.
4. *If you want to override the image for one WebWorks Help target*, create the WebWorks Help 5.0\Skins\theme\Files\wwhelp\wwhimpl\common\images folder in your *projectname***Targets** folder, where *theme* is the name of the WebWorks Help theme you want to override, such as Lobby_Blue, and *projectname* is the name of your ePublisher project. If the theme name is two words, include an underscore instead of a space between the words.
5. Copy the toolsbg.gif file from the following folder to the images override folder you created within your project folder:

Program Files\WebWorks\ePublisher Designer\Formats\WebWorks Help 5.0\ Skins\theme\Files\wwhelp\wwhimpl\common\images
6. Open the toolsbg.gif file you copied to your project override folder and modify it to be the background image of the toolbar pane.
7. Save and close the toolsbg.gif file.
8. Regenerate your project to review the changes.

Customizing the Navigation Pane in WebWorks Help

You can customize the navigation pane in several ways. For more information about the navigation pane, see “Navigation Pane in WebWorks Help”.

Setting the Initial Width of the WebWorks Help Navigation Pane

The navigation pane provides the Contents, Index, Search, and Favorites tabs. When the WebWorks Help opens, the initial width of the navigation pane is 300 pixels by default. You can override the `wwhelp.htm` file to define the initial width of the navigation pane.

To set the initial width of the navigation pane

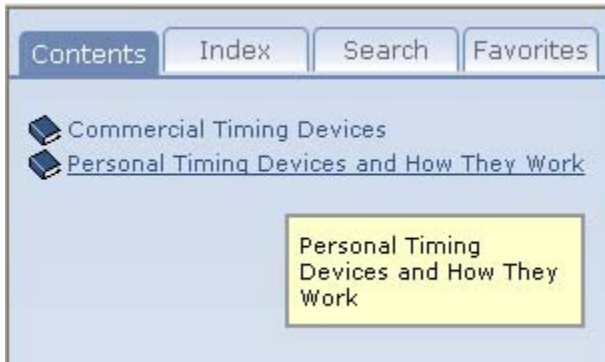
1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. *If you want to override the initial width for all WebWorks Help targets*, create the `Formats\WebWorks Help 5.0\Files\wwhelp\wwhimpl\js\html` folder in your *projectname* folder, where *projectname* is the name of your ePublisher project.
3. *If you want to override the initial width for one WebWorks Help target*, create the `Targets\WebWorks Help 5.0\Files\wwhelp\wwhimpl\js\html` folder in your *projectname* folder, where *projectname* is the name of your ePublisher project.
4. Copy the `wwhelp.htm` file from the following folder to the `html` override folder you created within your project folder:

`Program Files\WebWorks\ePublisher Designer\Formats\WebWorks Help 5.0\ Files\wwhelp\wwhimpl\js\html`
5. Open the `wwhelp.htm` file you copied to your project override folder.
6. Find the following line of code and modify the `300` to be the number of pixels wide you want the navigation pane to use as its initial width.

`<frameset cols="300,*" onLoad ...`
7. Save and close the `wwhelp.htm` file.
8. Regenerate your project to review the changes.

Controlling the Navigation Pane Hover Text Appearance

Hover text refers to the popup window that WebWorks Help displays when you mouseover (hover over) a link on the Contents tab, as shown in the following figure.



You can adjust the appearance of the hover text by overriding the `wwhelp_settings.xml` file. This file allows you to modify the font, font color, background color, and border color of the hover text and its popup window. You can also disable the hover text.

To change the navigation pane hover text appearance

1. Identify the theme in use for your WebWorks Help target that you want to modify. For more information, see “Selecting a Theme”.
2. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
3. *If you want to override the hover text settings for all WebWorks Help targets*, create the `Formats\WebWorks Help 5.0\Skins\theme` folder in your `projectname` folder, where `theme` is the name of the WebWorks Help theme you want to override, such as `Lobby_Blue`, and `projectname` is the name of your ePublisher project. If the theme name is two words, include an underscore instead of a space between the words.
4. *If you want to override the hover text settings for one WebWorks Help target*, create the `Targets\WebWorks Help 5.0\Skins\theme` folder in your `projectname` folder, where `theme` is the name of the WebWorks Help theme you want to override, such as `Lobby_Blue`, and `projectname` is the name of your ePublisher project. If the theme name is two words, include an underscore instead of a space between the words.
5. Copy the `wwhelp_settings.xml` file from the following folder to the `theme` override folder you created within your project folder:

Program Files\WebWorks\ePublisher Designer\Formats\WebWorks Help 5.0\ Skins\theme

6. Open the `wwhelp_settings.xml` file you copied to your project override folder.
7. Find the following block of code:

```
<HoverText enable="true">
...
</HoverText>
```

8. *If you want to disable the hover text*, replace `<HoverText enable="true">` with `<HoverText enable="false">`.

9. ***If you want to change the font family of the hover text***, replace `font-family: Verdana, Arial, Helvetica, sans-serif` with the names of the font family you want to use for the text. Make sure you specify fonts that are installed by default.
10. ***If you want to change the font size of the hover text***, replace `font-size: 8pt` with the size of the font you want to use for the text, such as `font-size: 10pt`.
11. ***If you want to change the font color of the hover text***, replace the color defined by `foreground="#000000"` with the RGB color value you want to use for the text.
12. ***If you want to change the background color of the popup window***, replace the color defined by `background="#FFFFFFCC"` with the RGB color value you want to use for the background.
13. ***If you want to change the border color of the popup window***, replace the color defined by `border="#999999"` with the RGB color value you want to use for the border.
14. Save and close the `wwhelp_settings.xml` file.
15. Regenerate your project to review the changes.

Changing the Font Color on the Navigation Pane Tabs in WebWorks Help

The tabs in the navigation pane all share the same font and text style properties. These style properties are specified in the `wwhelp_settings.xml` file.

To change the font color on all the tabs in the navigation pane

1. Identify the theme in use for your WebWorks Help target that you want to modify. For more information, see “Selecting a Theme”.
2. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
3. *If you want to override the tab font color for all WebWorks Help targets*, create the `Formats` \WebWorks Help 5.0\Skins\theme folder in your *projectname* folder, where *theme* is the name of the WebWorks Help theme you want to override, such as `Lobby_Blue`, and *projectname* is the name of your ePublisher project. If the theme name is two words, include an underscore instead of a space between the words.
4. *If you want to override the tab font color for one WebWorks Help target*, create the `Targets` \WebWorks Help 5.0\Skins\theme folder in your *projectname* folder, where *theme* is the name of the WebWorks Help theme you want to override, such as `Lobby_Blue`, and *projectname* is the name of your ePublisher project. If the theme name is two words, include an underscore instead of a space between the words.
5. Copy the `wwhelp_settings.xml` file from the following folder to the *theme* override folder you created within your project folder:

`Program Files\WebWorks\ePublisher Designer\Formats\WebWorks Help 5.0\ Skins\theme`
6. Open the `wwhelp_settings.xml` file you copied to your project override folder.
7. Find the following block of code:


```
<Tabs>
  <DefaultColors foreground="#666666" />
  <SelectedColors foreground="#FFFFFF" />
</Tabs>
```
8. *If you want to change the color of the text on the unselected tabs*, replace `#666666` with the RGB color value you want to use for the text on those tabs.
9. *If you want to change the color of the text on the selected tab*, replace `#FFFFFF` with the RGB color value you want to use for the text on that tab.
10. Save and close the `wwhelp_settings.xml` file.
11. Regenerate your project to review the changes.

Using Custom Icons on the Contents Tab in WebWorks Help

When you navigate through the table of contents on the Contents tab, you can see two types of icons for entries that have subentries. The open book icons (`fo.gif`) represent expanded table of contents levels, and the closed book icons (`fc.gif`) represent unexpanded table of contents levels, as shown in the following figure.



These icons are `.gif` images stored in the WebWorks Help theme folder. You can create override files for the `fo.gif` and `fc.gif` files to replace these files and use your own custom table of content icons instead.

To replace the book icons used on the Contents tab

1. Identify the theme in use for your WebWorks Help target that you want to modify. For more information, see “Selecting a Theme”.
2. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
3. *If you want to override the images for all WebWorks Help targets*, create the `WebWorks Help 5.0\Skins\theme\Files\wwhelp\wwhimpl\common\images` folder in your `projectname\Formats` folder, where `theme` is the name of the WebWorks Help theme you want to override, such as `Lobby_Blue`, and `projectname` is the name of your ePublisher project. If the theme name is two words, include an underscore instead of a space between the words.
4. *If you want to override the images for one WebWorks Help target*, create the `WebWorks Help 5.0\Skins\theme\Files\wwhelp\wwhimpl\common\images` folder in your `projectname\Targets` folder, where `theme` is the name of the WebWorks Help theme you want to override, such as `Lobby_Blue`, and `projectname` is the name of your ePublisher project. If the theme name is two words, include an underscore instead of a space between the words.
5. Copy the `fo.gif` and `fc.gif` files from the following folder to the `images` override folder you created within your project folder:

`Program Files\WebWorks\ePublisher Designer\Formats\WebWorks Help 5.0\ Skins\theme\Files\wwhelp\wwhimpl\common\images`
6. Open the `fo.gif` and `fc.gif` files you copied to your project override folder and modify these files to be the book icons you want to use.
7. Save and close the `fo.gif` and `fc.gif` files.
8. Regenerate your project to review the changes.

Modifying the Appearance of the Search Message in WebWorks Help

In WebWorks Help, when users click on the Search tab, the message “Type in the word(s) to search for:” is displayed. Once the user types in a word to search for and clicks **Go**, the “(Searching)” message is briefly displayed in the search results box. The appearance of these messages is controlled by the `h2` and `h3` HTML tags, and these tags are located in the `search.js` file. To modify the appearance of the search message, you need to specify the style properties for `h2` and `h3` tags in a separate `.css` file.

Note: *If you customize the appearance of the search message by modifying the `search.js` file*, you will be responsible for maintaining your customizations as needed each time you update your Stationery to work with a new version of ePublisher.

For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.

To change the appearance of the “Type in the word(s) to search for” message, create a style definition for the `h2` tag. To modify the appearance of the “(Searching)” message, create a style definition for the `h3` tag. After you specify the font properties in a `.css` file for the `h2` and `h3` tags, reference the `.css` file in the `search.js` file.

Note: The `messages.xml` file in the `Formats\WebWorks Help 5.0\Files\wwhelp` folder controls the message text used throughout WebWorks Help for multiple languages. You can override this file to provide customized messages for your specific needs.

To modify the appearance of the search messages

1. *If you want to override the message appearance for all WebWorks Help targets*, complete the following steps:

- a. In your Stationery design project, on the **View** menu, click **Format Override Directory**.
- b. Create the following folder in your `projectname` folder, where `projectname` is the name of your ePublisher project:

```
Formats\WebWorks Help 5.0\Files\wwhelp\wwhimpl\js\scripts
```

2. *If you want to override the message appearance for one WebWorks Help target*, complete the following steps:

- a. In your Stationery design project, on the **View** menu, click **Target Override Directory**.
- b. Create the following folder in your `projectname` folder, where `projectname` is the name of your ePublisher project:

```
Targets\WebWorks Help 5.0\Files\wwhelp\wwhimpl\js\scripts
```

3. Copy the `search.js` file from the following folder to the `scripts` override folder you created within your project folder:

```
Program Files\WebWorks\ePublisher Designer\Formats\WebWorks Help 5.0\Files\wwhelp\wwhimpl\js\scripts
```

4. Open the `search.js` file you copied to your project override folder.
5. Find the following block of code:

```
// Display initializing message  
//
```

```
HTML.fAppend("<h2>" + WWHFrame.WWHJavaScript.mMessage.mInitializingMessage + "</h2>\n");
```

6. Replace that block of code with the following code:

```
// Display searching message
//

HTML.fAppend("<h2 style=\"color:#FF00FF; text-decoration: underline; font-size: 12pt;
\">" + WWHFrame.WWHJavaScript.mMessages.mSearchSearchingMessage + "</h2>\n");
```

7. Find both instances of the following block of code:

```
// Display searching message
//

HTML.fAppend("<h2>" + WWHFrame.WWHJavaScript.mMessages.mSearchSearchingMessage + "</h2>
\n");
```

8. Replace each instance of that block of code with the following code:

```
// Display searching message
//

HTML.fAppend("<h2 style=\"color:#FF00FF; text-decoration: underline; font-size: 12pt;
\">" + WWHFrame.WWHJavaScript.mMessages.mSearchSearchingMessage + "</h2>\n");
```

9. Find the following block of code:

```
// Display search message and/or prepare results for display
//
if (this.mSavedSearchWords.length == 0)
{
HTML.fAppend("<h3>" + WWHFrame.WWHJavaScript.mMessages.mSearchDefaultMessage + "</h3>
\n");
}
```

10. Replace that block of code with the following code:

```
// Display search message and/or prepare results for display
//
if (this.mSavedSearchWords.length == 0)
{
HTML.fAppend("<h3 style=\" color:#FF00FF; text-decoration: none; font-size: 14pt;\">" +
WWHFrame.WWHJavaScript.mMessages.mSearchDefaultMessage + "</h3>\n");
}
```

11. Save and close the `search.js` file.

12. Regenerate your project to review the changes.

Modifying the Search Ranking

The search results are displayed in the Search tab when a user types a word to search for and clicks **Go**. The search results are sorted by the relevancy ranking, which is calculated based on the scoring preference defined for the HTML tags in the `wwhelp_files.xml` file. By default, WebWorks Help assigns relevancy rankings based on where in a topic a particular item is found.

For example, if you set the scoring preference, or weight, for Heading 1 to 25 and you set the weight for Heading 2 to 15, then any search term found in a Heading 1 returns a higher relevancy ranking than if the search term is found in a Heading 2.

The following scenario illustrates how the relevancy ranking is calculated:

If you search for the word `popup`, and that word appears in both a Heading 1 and a Heading 2 on one page, and in a Heading 1 on another page, the search returns two results sorted by the relevancy ranking.

To get the relevancy ranking, WebWorks Help takes the score of each search result and divides it by the highest score found. To get the score of each search result, the scoring preference for each HTML tag is used.

For example, Heading 1 has a scoring preference of 25 and Heading 2 has a scoring preference of 15. If the word “popup” appears in a Heading 1 and a Heading 2 on a topic page, then the score for that word on that page is $25+15=40$. If the word “popup” appears in a Heading 1 on another topic page, then the score for that word on that page is 25. Therefore, if 40 is the highest score that a search for “popup” returns, the relevancy ranking of the first search result is $100\%=40/40\times 100\%$, and the relevancy ranking for the second search result is $38\%=15/40\times 100\%$.

To modify the relevancy ranking for search results

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. *If you want to override the message appearance for all WebWorks Help targets*, create the `Formats\WebWorks Help 5.0\Transforms` folder in your `projectname` folder, where `projectname` is the name of your ePublisher project.
3. *If you want to override the message appearance for one WebWorks Help target*, create the `Targets\WebWorks Help 5.0\Transforms` folder in your `projectname` folder, where `projectname` is the name of your ePublisher project.
4. Copy the `wwhelp_files.xml` file from the following folder to the `Transforms` override folder you created within your project folder:

```
Program Files\WebWorks\ePublisher Designer\Formats\WebWorks Help 5.0\ Transforms
```

5. Open the `wwhelp_files.xml` file you copied to your project override folder.
6. Find the following block of code:

```
<ScoringPrefs>
  <meta name="keywords" weight="100"/>
  <meta name="description" weight="50"/>
  <meta name="summary" weight="50"/>
  <title weight="20"/>
  <h1 weight="15"/>
  <h2 weight="10"/>
  <caption weight="10"/>
  <h3 weight="7"/>
  <th weight="5"/>
```

```
<h4 weight="5"/>
<h5 weight="4"/>
<h6 weight="3"/>
<h7 weight="2"/>
</ScoringPrefs>
```

7. Modify the weight attributes for any tags, such as `h1` and `h2`, you want to change.
8. Save and close the `wwhelp_files.xml` file.
9. Regenerate your project to review the changes.

Modifying the Search Highlighting

You can control search highlighting with an override:

```
Formats\WebWorks Help 5.0\Skins\[skin name]\wwhelp_settings.xml
```

For more information regarding overrides, See “Creating Format Overrides”.

The following markup controls the behavior:

```
<Search enable="true">
<Results showrank="true" />
<Highlighting enable="true">
  <Colors foreground="#FFFFFF" background="#333399" />
</Highlighting>
</Search>
```

Using an HTML value for the color will help you control the search highlighting color for the text and for the background of the word that is being searched in the output page.

Note: This browser behavior is limited to Internet Explorer.

Synonyms

By defining a list of synonyms, you can increase the effectiveness of a search by grouping similar words together as a single result. For example, you define movie, video, and avi as synonyms. If a user searches for one of those terms, a match will be scored if any of the three are found in a document.

To add these definitions, you'll need to perform an override on a file called `synonyms.xml`, located by default in the ePublisher Designer installation directory here:

```
\Formats\WebWorks Help 5.0\Files\wwhdata\common
```

Once the file is copied to the correct location in your project directory, open it in any text- or XML-editing application (e.g., NotePad), and add entries to the `<WebWorksSynonyms>` section. For example, your synonyms might look like this:

```
<WebWorksSynonyms>
  <Word value="movie">
    <Synonym value="avi" />
    <Synonym value="video" />
  </Word>
  <Word value="picture">
    <Synonym value="image" />
    <Synonym value="graphic" />
    <Synonym value="photo" />
  </Word>
</WebWorksSynonyms>
```

Please note that the Synonyms feature does not recognize multiple word values or synonyms. And, it ignores words (and synonyms) of 2 letters or less. Also, its search logic is quite literal. That is, it finds only what you type, and only if the exact form is in the text. For example, it does not find plurals or phrases, and it even includes punctuation.

Minimum word length & common words

When performing a text search, the WebWorks Help 5.0 search engine follows a couple of guidelines designed to make the search more efficient and the results more helpful. By default, the minimum length of a word in the search results is 3 letters. ePublisher Designer also comes with a list of terms which appear commonly in many types of documents, and are therefore left out of search results. You can modify both the minimum length and the list of common words by performing a target override on a file called `locales.xml`, located by default in the ePublisher Designer installation directory, here:

```
2018.1\Formats\Shared\common\locale
```

For tips on performing the override, See “Creating Format Overrides”..

Because this file is in the Shared formats directory, your project folder hierarchy should look like this:

```
MyProject\Targets\MyWWHelpTarget\Shared\common\locale
```

To change the minimum number of letters in a WebWorks Help 5.0 search result:

1. Open `locales.xml` in an XML- or text-editing application (e.g., NotePad).
2. Find the Locale used by your project (default English is “en”).
3. Under the `<Search>` section, change the `MinimumWordLength` value to the desired number of characters in the shortest word the search should return.

To change the list of words which will not be returned in a WebWorks Help 5.0 search:

1. Open `locales.xml` in an XML- or text-editing application (e.g., NotePad).
2. Find the Locale used by your project (default English is “en”).
3. Under the `<Search>` section, modify the `<StopWords>` values to reflect the list of words which should not appear in your project’s search results.

Using Context-Sensitive Help in WebWorks Help

Context-sensitive help links provide content based on the context of what the user is doing. In many cases, this help content is based on the window that is open and active. For example, the **Help** button on a window in a software product can open a specific help topic that provides important information about the window and links to related topics.

WebWorks Help allows you to use a TopicAlias marker to define an internal identifier for each topic. The benefit of using an internal identifier is that it allows file names to change without impacting the links from the product. The writer inserts this marker in a topic and specifies a unique value for that topic. ePublisher creates a mapping file to identify each topic associated with a unique value. Then, WebWorks Help uses this internal identifier and the mapping file to display the correct topic. Before you can reference topics in WebWorks Help using topic aliases, you must enable TopicAlias markers in your Stationery. For more information, see “Defining Filename Markers for Context-Sensitive Help Links”.

Understanding Mapping Files in WebWorks Help

WebWorks Help does not generate a mapping file. However, you can see a list of defined contexts and topics in the `wdata/xml/files.xml` file. You can also use the Topics Report to verify that context-sensitive help topics have been created for each topic ID specified in your source document. The Topics Report lists the topic ID and the topic file created for each topic ID. Topic aliases report. For more information about the topics report, see “Understanding Topics Reports”.

Opening Context-Sensitive Help in WebWorks Help using Standard URLs

You can open WebWorks Help from the application using standard URLs or the WebWorks Help API. For more information about the API, see “Opening Context-Sensitive Help with the WebWorks Help API”.

Note: If you have WebWorks Help installed on the local computer, Internet Explorer 7 ignores the query string attached to a URL when it reloads a page. For example, after displaying a security warning or other message, Internet Explorer reloads the page. Without the query string, the browser cannot open the specific topic, since the group name and topic alias are not available. To avoid this issue with Internet Explorer 7, you can use the WebWorks Help API to open a topic in locally installed WebWorks Help.

To open the complete WebWorks Help output to the default help topic, open the `index.htm` or `index.html` file in the root of the folder where the WebWorks Help is stored. This file defines the help frameset and loads the individual components, such as the table of contents, index, and topic content.

To open a specific topic in WebWorks Help, use the following URL:

```
helplocation/wwhelp/wwhimpl/api.htm?context=groupname&topic=alias
```

The variable parts of this URL are defined as follows:

helplocation

Specifies the location where the WebWorks Help is installed. If the help is on a Web server, specify the location using the `http` protocol and the Web site path to the root of the help, such as `http://www.webworks.com/help`. If the help is installed on the local computer, specify the location using the `file` protocol and the path to the root of the help, such as `file:///C:/Program Files/YourApplication/help`.

groupname

Specifies the group context value for the top-level group in which the topic resides in Document Manager. This group context is specified in merge settings for each top-level group.

alias

Specifies the value of the TopicAlias marker in the topic to open.

Opening Context-Sensitive Help with the WebWorks Help API

WebWorks Help provides an API that you can use to call context-sensitive topics in WebWorks Help. Using the API instead of simple URLs gives you increased flexibility and enhanced control. For example, the following list highlights some of the benefits of using the API:

- The API automatically determines the default browser and opens it to display the correct help topic.
- The API reduces the amount of code developers must write to integrate context-sensitive help. Without the API, developers must code their own COM interface to communicate with the browser on the Windows platform. This code is provided as part of the WebWorks Help API.
- Using the API can significantly reduce the time required to load an individual topic if the help system is already open in the browser. If your application calls a topic using a URL instead of the API, the entire frameset, including the WebWorks Help applet, is loaded each time the user opens help. If the application calls a topic using the API, and the correct help is already loaded in the browser, neither the frameset nor the applet is reloaded. Instead, the currently open topic pane displays the correct help topic, which delivers a significant performance improvement for WebWorks Help users.
- The API allows you to avoid the issue that exists with standard URLs in Internet Explorer 7. In Internet Explorer 7, if you have WebWorks Help installed on the local computer, Internet Explorer 7 ignores the query string attached to a URL when it reloads a page. For example, after displaying a security warning or other message, Internet Explorer reloads the page. Without the query string, the browser cannot open the specific topic, since the group name and topic alias are not available.

You can use the C/C++ API, which is available as a .dll or a COM object. This API supports WebWorks Help 4.0 and 5.0. WebWorks Help 5.0, does not include the Java navigation. Therefore, do not use the Java navigation options with WebWorks Help 5.0. For more information about this API and the software development kit, see wiki.webworks.com/DevCenter/Projects/WebWorksHelp/WebWorksHelpSDK.

Opening Context-Sensitive Help with the Javascript API

Similar to using the WebWorks Help API, you can use the Javascript API when working with web applications designed to run on websites using standard HTML and Javascript. For complete details of the WebWorks Help Javascript API, see wiki.webworks.com/DevCenter/Projects/WebWorksHelp/JavascriptAPI.

Customizations Specific to WebWorks Reverb

You can customize the appearance and behavior of WebWorks Reverb in several ways. For example, you can customize the colors and images, and toolbar buttons displayed in the browser. The following sections describe these WebWorks Reverb-specific customizations. For more information about changes you can make to multiple output formats, see “Checklist: Design, Deploy, and Manage Stationery”.

Changing the Appearance of WebWorks Reverb

Most customizations to WebWorks Reverb rely upon changing one of three file types:

- *Page Templates* - Found in the Pages folder and have the extension .asp. Users may create overrides for these files in their project to change overall layouts
- *skin.css* - Found in the Pages\css folder. Users may change the color, icons, and other layout specific behaviors of the skin with this file. The skin settings also include the layout behaviors of breadcrums and the company information that appears above the toolbar. Changes to this file, will not have an affect on the content pages.
- *webworks.css* - Found in the Pages\css folder. This file is very similar to the skin.css file except that it only controls the layout behaviors of the content pages. This includes the mini-TOC and Related Topic styling behaviors.
- *skin.png* - Found in the Pages\images folder. This file works as a CSS sprite file, which means that it contains multiple images. Users can quickly modify colors within this file to meet branding requirements.

In the Pages folder there are sample files that can be used to preview customizations to the skin.css, webworks.css, and skin.png files. After making changes to any of these files, load the Skin.html file directly into a browser and view all aspects of the Reverb skin.

For easy reference, here are a list of page template files and their corresponding output files:

Page Template	Use
Connect.asp	Entry point to the Reverb run-time. This file controls the placement of the toolbar buttons, table of contents/index panels
Parcel.asp	Data file for each Reverb parcel (help set)
Index.asp	Data file for each Reverb parcel's Index information
Splash.asp	Defines the layout and appearance of the Reverb splash page
Page.asp	Defines the layout and appearance of generated pages
Search.asp	Search integration page

Finally, WebWorks Reverb makes use of CSS classes to identify elements which should perform special actions. These elements may occur in the entry page (default to index.html) or in generated pages.

CSS Class	Related Action
ww_behavior_toc	Displays merged TOC
ww_behavior_index	Displays all defined Indexes
ww_behavior_search	Displays search system
ww_behavior_globe	Enables/disables Google Translate
ww_behavior_home	Navigates to the first page in the browse sequence

CSS Class	Related Action
ww_behavior_prev	Navigates to the previous page in the browse sequence
ww_behavior_next	Navigates to the next page in the browse sequence
ww_behavior_print	Prints the current page
ww_behavior_email	Emails feedback on the current page URL
ww_behavior_pdf	Display related PDF for the current page

Format Settings for WebWorks Reverb

WebWorks Reverb defines several format specific settings.

WebWorks Reverb Format Settings

The complete reference of Reverb Settings.

Format Setting	Use
Display large images in lightbox	When a thumbnail is used for an image, the full size version of the image can be viewed by clicking the image which is then displayed in a lightbox. When disabled the image displays in a separate file.
Enable Print Icon	Enables/disables the print icon in generated pages
Entry Filename	Specifies the name of the Reverb entry-point file name (default is "index.html")
Feedback Email	Defines the feedback email address for use in generated pages
Feedback Email Message	Defines the contents of the feedback email subject line and body section. Use \$Location; to insert the URL of the current page
Home	Enables/disables the home navigation button which navigates users to the first page in the browse sequence.
Minimum height for non-scrolling toolbar	Specifies the minimum browser window height required for Reverb to fix the toolbar at the top of the window instead of scrolling out of view.
Minimum page width for sidebar	Specifies the minimum browser window width required for Reverb to display a sidebar TOC/Index next to the content page. If the window width is less than this value then the TOC/Index will displayed in place of the content panel and only when the TOC/Index button is selected.
Reverb Page Style	Specifies the page style to use when processing the Reverb entry-point file that encapsulates the TOC, Index, Search, and content panels.
Search Implementation	Specifies the type of search engine to use by Reverb
Sidebar width	Specifies the width of the sidebar TOC/Index when displayed next to the content area
Skin	Specifies an alternate plugin file to use for changing the look-and-feel of the Reverb skin.
Splash Page Style	Specifies the page style to use when processing the splash page
Use first document as splash page	Determines initial page displayed
WebWorks Help API Compatibility	Enables/disables compatability with the WebWorks Help API allowing Reverb to replace existing implementations of WebWorks Help while using the same API.

Google Format Settings

The complete reference of Google Settings.

Format Setting	Use
Google Analytics Identifier	Enables the use of Google Analytics to measure page activity
Google Translate	Enables support for the Google Translate web service

Social Format Settings

The complete reference of Social Media Settings.

Format Setting	Use
Disqus Identifier	Enables support for user comments using the Disqus comment web service
Disqus - Allow non-public networks	Enable users to post Disqus comments behind a firewall or non-public website
FaceBook Like	Enables users to report "I like this!" in Facebook
Google +1 Button	Enables users to recommend the current page to their Google +1 social networks
LinkedIn Share	Enables users to share the current page to their LinkedIn community as well as increment the counter that shows how many times the link has been shared.
Tweet This!	Enable Twitter "Tweet This!" support

Selecting an Alternate Skin for WebWorks Reverb

In WebWorks Reverb, you can select among several alternate skins to define the appearance of your help. When you specify an alternate skin, the underlying ASP and CSS files will be automatically updated to reflect the differences in that skin.

To choose an alternate skin for WebWorks Reverb

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. In the **WebWorks Reverb** category, select the right column of the **Skin** entry to display the file picker button.
4. Click the file picker button to bring up an **Open** file dialog which will display a list of skin plugin files. Each skin plugin file is identifiable by a `.weplugin` extension.
5. Browse to the plugin file that you wish to use and double-click it to set the skin to that value.

Once you have set an alternate skin, you can later change it again using the same procedure.

Warning: If you have customized any files using the **Advanced** menu to create a Target override, then you will need to remove that customization and then re-implement it again after you change the skin setting. For more information on implementing and managing Target overrides see “Creating Target Overrides”.

Note: If you are going to customize an alternative Reverb skin, first set the skin type in the **Target Settings** then create a **Target Customization** override for the file(s) that you want to customize.

Customizing the Top-Level Entry File

Specifying the Entry Page Name

When you generate output for your project, a top-level entry file is created. This file defines the frameset for the help and gives the user a file to open that displays the complete help. By default, the top-level entry file is named `index.html`, but you can specify any name you need for the top-level entry file as long as you specify an `.html` or `.htm` extension.

To rename the top-level file

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. In **Entry filename**, type the name you want to use for the top-level entry file.
4. Click **OK**.
5. Regenerate the project and review the results.

Specifying the Entry Page Style

WebWorks Reverb allows users to leverage ePublisher's Style Designer to set color and layout properties of the generated entry page. Users can specify a page style to use under the WebWorks Reverb section of the Format Settings dialog.

Customizing the Splash Page in WebWorks Reverb

You can modify or remove the splash page that is displayed while your WebWorks Reverb opens. You can customize the splash page in the Stationery, and then writers can override this customization in each project, as needed.

Specifying the Splash Page Style

WebWorks Reverb allows users to leverage ePublisher's Style Designer to set color and layout properties of the generated splash page. Users can specify a page style to use under the WebWorks Reverb section of the Format Settings dialog.

Replacing the Splash Image

The splash page is the first page that displays in the topic pane when the help set launches initially. By default, WebWorks Reverb systems display the WebWorks Reverb splash image. You can replace the default splash image with a custom image.

To replace the splash page image

1. Identify the theme in use for your WebWorks Reverb target that you want to modify. For more information, see “Selecting a Theme”.
2. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
3. *If you want to override the image for all WebWorks Reverb targets*, create the **Formats**WebWorks Reverb \Pages\images folder in your *projectname* folder, where *projectname* is the name of your ePublisher project.
4. *If you want to override the image for one WebWorks Reverb target*, create the **Targets**WebWorks Reverb \Pages\images folder in your *projectname* folder, where *projectname* is the name of your ePublisher project.
5. Copy the `splash.jpg` file from the following folder to the `images` override folder you created within your project folder:

```
Program Files\WebWorks\ePublisher Designer\Formats\WebWorks Reverb\ Pages\images
```
6. Open the `splash.jpg` file you copied to your project override folder and modify it to be the splash page image you want.
7. Save and close the `splash.jpg` file.
8. Regenerate your project to review the changes.

Modifying the Splash Page

Users may also create an override for the splash page template, `Splash.asp`. This allows users to change every aspect of a splash page's appearance.

Removing the Splash Page

When WebWorks Reverb opens, it displays the splash page. However, instead of displaying the splash page, you can configure WebWorks Reverb to display the first topic in the help.

To remove the splash page

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. Set **Use first document as splash page** to **Enabled**.
4. Click **OK**.
5. Regenerate the project and review the results.

Using Context-Sensitive Help in WebWorks Reverb

Context-sensitive help links provide content based on the context of what the user is doing. In many cases, this help content is based on the window that is open and active. For example, the **Help** button on a window in a software product can open a specific help topic that provides important information about the window and links to related topics.

WebWorks Reverb allows you to use a TopicAlias marker to define an internal identifier for each topic. The benefit of using an internal identifier is that it allows file names to change without impacting the links from the product. The writer inserts this marker in a topic and specifies a unique value for that topic. ePublisher creates a mapping file to identify each topic associated with a unique value. Then, WebWorks Reverb uses this internal identifier and the mapping file to display the correct topic. Before you can reference topics in WebWorks Reverb using topic aliases, you must enable TopicAlias markers in your Stationery. For more information, see “Defining Filename Markers for Context-Sensitive Help Links”.

Understanding Mapping Files in WebWorks Reverb

WebWorks Reverb does not generate a mapping file. However, you can see a list of defined contexts and topics in the parcel file. You can also use the Topics Report to verify that context-sensitive help topics have been created for each topic ID specified in your source document. The Topics Report lists the topic ID and the topic file created for each topic ID. Topic aliases report. For more information about the topics report, see “Understanding Topics Reports”.

Opening Context-Sensitive Help in WebWorks Reverb using Standard URLs

You can open WebWorks Reverb from the application using standard URLs.

To open a specific topic in WebWorks Reverb, use the following URL:

```
helplocation/index.html#context/group_context/topic_alias
```

The variable parts of this URL are defined as follows:

helplocation

Specifies the location of the desired WebWorks Reverb help set. If the help is on a Web server, specify the location using the `http` protocol and the Web site path to the root of the help, such as `http://www.webworks.com/help`.

group_context

Specifies the group context value for the top-level group in which the topic resides in Document Manager. This group context is specified in merge settings for each top-level group.

topic_alias

Specifies the value of the TopicAlias marker in the topic to open.

URL Commands Support by WebWorks Reverb

WebWorks Reverb supports additional commands in addition to context-sensitive help.

Command	Related Action
index.html# context / <i>group_context</i>	Display specified context-sensitive help topic
index.html# page / <i>child_page_page_id</i>	Displays define child page
index.html# search / <i>search words</i>	Initiates search for specific terms
index.html# toc /	Display the table of contents panel
index.html# index /	Display the index panel

Opening Context-Sensitive Help in WebWorks Reverb using Javascript

You can use javascript to open your context-sensitive help links when working with web applications or websites designed with HTML and javascript. For complete details of using WebWorks Reverb with web applications and websites, see [wiki.webworks.com/DevCenter/Projects/Reverb/For Web Applications](http://wiki.webworks.com/DevCenter/Projects/Reverb/For%20Web%20Applications).

Opening Context-Sensitive Help in WebWorks Reverb using the WebWorks Help API

In addition to using simple URLs for opening context-sensitive help, you can alternatively use an API that is incorporated into your application. Using an API makes it easier to enable help buttons in your application so that the load the correct page in your Reverb output.

WebWorks Reverb uses the same API as WebWorks Help 4 and 5. For information about the benefits and steps for using this API, see “Opening Context-Sensitive Help with the WebWorks Help API”.

Steps for Enabling the Context-Sensitive Help API for use with WebWorks Reverb

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. In the **WebWorks Reverb** category, set **WebWorksHelpAPICompatibility** to **Enabled**.
4. Click **OK** to save the target settings.
5. Have your application developers download the WebWorks Help SDK located at: wiki.webworks.com/DevCenter/Projects/WebWorksHelp/WebWorksHelpSDK. Depending on the technologies your application uses, your developers will be able to incorporate the API and then make context-sensitive API calls to your published Reverb output.

Configuring Client-Side Search for Reverb

This is the default search configuration for WebWorks Reverb in the Target Settings. This search implementation requires no additional setup or configuration. The search implementation supports both wildcard (*) and phrase searching.

Configuring Synonyms

By defining a list of synonyms, you can increase the effectiveness of a search by grouping similar words together as a single result. You can define as synonyms words that are not in your source documents, but the main word does need to be in a source file. For example, if you define `video` and `avi` as synonyms of `movie` (main word), then when a user searches for `video`, a match will be scored showing the source files where the word itself is in, but also where the main word `movie` is as well. Please note that the inverse is not true, meaning that if a user searches instead for `movie`, files where the word `video` is will not be scored because of the synonym entry.

To add these definitions, you'll need to perform an override on a file called `locales.xml`, located by default in the ePublisher Designer installation directory here:

```
\Formats\WebWorks Reverb\Shared\common\locale
```

Note: For Reverb 2 substitute `WebWorks Reverb` for `WebWorks Reverb 2.0`.

Once the file is copied to the correct location in your project directory, open it in any text- or XML-editing application (e.g., NotePad), and add entries to the `<Synonyms>` section. For example, your synonyms might look like this:

```
<Synonyms>
  <Word value="movie">
    <Synonym value="avi" />
    <Synonym value="video" />
  </Word>
  <Word value="designer">
    <Synonym value="pro" />
    <Synonym value="publisher" />
  </Word>
</Synonyms>
```

Please note that the Synonyms feature does not recognize multiple word values or synonyms. And, it ignores words (and synonyms) shorter in length than the specified value for `MinimumWordLength` defined in the same file. Synonyms are not case sensitive.

Configuring Microsoft IIS Search for Reverb

If you are deploying Reverb to a private network and require a more advanced search capability than provided by the *Client-side Search* implementation, then the *Microsoft IIS Search* implementation would be a good choice.

In order to choose this option for your Reverb search implementation, you must have a Microsoft IIS web server to host the Reverb output files.

To index the files in the IIS search, complete the following steps

1. In order for the Microsoft IIS search solution to work properly, you will need to make sure that your project's ePublisher target setting has **Pretty Print** enabled. This target setting is located under the **File Processing** group.
2. Generate and then deploy your content to a folder served by your Microsoft IIS server.
3. Right-click on the target deployment folder.
4. Go to **Properties** -> **Advanced**
5. Check the option to **Allow Files on this drive to have contents indexed in addition to file properties**.
6. Click **OK**, and then click **OK** again.
7. For running on Windows Server, you will need to specify the **Indexing Options** on the machine as follows.
 - a. In Windows Server, open **Control Panel > Indexing Options**.
 - b. Click on the **Advanced** button and then switch to the **File Types** tab.
 - c. Select **Index Properties and File Contents**.
 - d. Click **OK**, and then click **OK** again.
8. Access your Reverb output via a web browser to verify search is working.

For the latest information and configuration steps see:

<http://wiki.webworks.com/Permalinks/Solutions/Output/WebWorksReverb/ConfiguringIIS/ConfiguringSearch/>.

Searching WebWorks Reverb - URL Method

When viewing the URL for your help, you can use a search term after the URL, for example you can go to the link here: <http://www.webworks.com/Documentation/Reverb/#search/ePublisher>

When clicked, the link above searches for the term “ePublisher” this can be changed to any search term, which will produce results accordingly.

Incorporating Google Analytics for Your Reverb Files

WebWorks Reverb is not only a fully functional help system for your content, it's also a collection of distinct web page files that can be viewed individually on a website. In fact, you can very easily share URLs of individual page files from your Reverb help system. These URLs will load just like a single web page within a browser. While this is very nice from an end-user perspective, because the load times are very minimal, it is even better from a publisher perspective. Simply put, you can now easily track the usage patterns and frequencies of all your Reverb help pages, which is where the value of Google Analytics comes in; all you have to do is either determine the appropriate Google *Analytics Account ID* that your website is currently using (if Google Analytics has already been setup for your website), or create a Google Analytics Account which will then provide a unique identifier for you website or website path. Since the 2011.1 version of WebWorks Reverb, you can follow user search patterns with the Google Analytics' Site Search feature. This allows you to review search terms and search start pages for your content. For more information, please refer to the following page:

<http://analytics.blogspot.com/2007/11/site-search-now-available.html>

Once you have obtained the appropriate Google *Analytics Account ID*, then you simply set your ePublisher project's format property called: `Google Analytics Identifier`.

Steps to Create a Google Analytics Account

1. Visit the URL: <http://www.google.com/analytics>. From here you will create an account for your particular domain, sub-domain, or URL website path.

Note: Google Analytics allows you to create individual identifiers for separate domains (i.e. www.webworks.com), sub-domains (i.e. www.webworks.com), or website paths (i.e. www.webworks.com/Documentation). For most businesses and organizations you have a single account identifier for each sub-domain and that value is used by all the web pages on that sub-domain. Please note that network firewalls can prevent tracking from occurring. For more information please refer to the following page: <http://wiki.webworks.com/Permalinks/Solutions/Output/WebWorksReverb/GoogleAnalyticsAndFirewalls/>.

2. *If you have not previously accessed Google Analytics*, then select the link labeled: **Sign Up Now**. otherwise simply select the **Access Analytics** button.

Note: You are required to have a Google account when working with any Google service.

3. In the *Accounts* table, displayed on the *Overview: all accounts* page, select the link labeled: **+ Add new account**.

4. Select the **Sign Up** button and complete the next form.

- a. In the field: *Website's URL*, specify your website's URL (i.e. www.webworks.com) where you will be deploying your Reverb help files.

- b. Specify both your time zone and country in the remaining fields.

Note: The time zone values are important because this determines the day boundary for your accumulated statistics. Most often you compare website traffic over a specified period of days, so what time zone you select for the account may impact how the values get calculated.

- c. Select the **Continue** button to proceed to the *Contact Information* page.

- d. Fill out the *Contact Information* and then select the **Continue** button to proceed to the *Accept User Agreement* page.

- e. Finally, select the **Create New Account** button and your account will be created. After you create the account you can modify the settings that you have provided in case something changes later.

Steps to Obtain Your Google Analytics Account Identifier

1. Visit the URL: <http://www.google.com/analytics>. From here you will access the account used to manage or view your website profile.
2. *If someone has already created a Google Analytics Account for your website*, then you will need to have an *Administrator* user of the account grant you access to a specific *Website Profile*, which can restrict or grant access to certain functionality in Google Analytics.

Note: When an *Administrator* user grants you access to an Analytics Account, they will need your Google Account email address, which will then become the login you will use to access the analytics account.

3. You may have access to more than one Analytics Account, so next, choose your account by selecting the link labeled with the correct account name. The *Website Profiles* available for this account when then be displayed.
4. In the table labeled: *Website Profiles*, you will see the *Google Analytics Account Identifier*. All account identifiers will have the pattern: **UA-XXXXXXXX-Y** (i.e. UA-18761398-1). This is the value you will use for the ePublisher project format setting called: `Google Analytics Identifier`.

Google Analytics behind a Firewall or Non-public Website

According to the Google documentation, In order for Google Analytics to generate reports for your corporate intranet usage, your corporate network must be able to reach the ga.js JavaScript file at <http://www.google-analytics.com/ga.js> or <https://www.google-analytics.com/ga.js>.

If you can reach the above URL using your company Internet connection, you have satisfied the first requirement. Additionally, your intranet must be accessible through a fully qualified domain name such as <http://intranet.example.com>. The ga.js JavaScript will not work if your intranet can only be accessed using a domain name that is not fully qualified, such as <http://intranet.example.com>.

For the latest information on this subject see: <http://wiki.webworks.com/Permalinks/Solutions/Output/WebWorksReverb/GoogleAnalyticsAndFirewalls/>

Configuring Commenting and End-User Feedback for Reverb

WebWorks Reverb uses the *Disqus* commenting platform for enabling a very powerful and cost-effective commenting and discussion system. The integration with WebWorks Reverb is not only transparent, it is optimized to ensure fast downloads. To enable this feature, all you need to do is set up a Disqus account and then create a *Disqus Site* for your Reverb help volume(s). Once you create a *Disqus Site*, you will then use the site's *Disqus Site Shortname* as the setting value for your ePublisher project's format setting called: `Disqus Identifier`. The Disqus Site can then be used to track and manage all the threads used within your deployed Reverb help volume(s). For the reports, you can use the Disqus API as offered in the Plus addon: <http://disqus.com/addons>. For more information regarding the API, refer to the documentation: <http://disqus.com/api>

Steps to Create Your First Disqus Site

1. Visit the URL: <http://disqus.com/>. Then select the **Sign Up** button. You will then be taken through a few simple forms which will create your *Disqus Login* as well as your initial *Disqus Site*.
2. In the field `Site URL`, specify the URL that best describes where your Reverb volume(s) will exist on your website.

Note: You can use a single Disqus site for all your Reverb deployments that exist on the same website or you can use separate Disqus sites for each separately deployed Reverb volume. Disqus provides a wide range of solutions for managing and moderating threads so choosing either strategy will be effective and depends more on how your organization's process will be handled.
3. In the field `Site Name`, specify a unique, human readable name that best describes the site. Then check the `Site Shortname` text box field and correct it to suit your preference. The *Site Shortname* will not only be used by ePublisher, but will also become the sub-domain used on Disqus. So if your *Site Shortname* is *epub*, then the URL used for this site's threads will be <http://epub.disqus.com>.
4. On the same page, specify the information for the *Primary Moderator*, other moderators can be added later. Then select the **Continue** button.
5. On the Settings page, specify your language and any optional features that you wish to also use. Then select the **Continue** button. You now have a *Disqus Login* as well as your first *Discus Site*. No additional steps are required to begin using your new site

Customizations Specific to WebWorks Reverb 2.0

WebWorks Reverb 2.0 uses SASS technology to build a robust and responsive layout. Using ePublisher's implementation of this technology, the user can customize any aspect of the visual layout. For more information about changes you can make to multiple output formats, see "Checklist: Design, Deploy, and Manage Stationery".

Changing the Appearance of WebWorks Reverb 2.0

Most customizations to WebWorks Reverb 2.0 rely upon changing one of three file types:

- *Page Templates* - Found in the Pages folder and have the extension .asp. Users may create overrides for these files in their project to change overall layouts.
- *SASS Files* - Found in the Pages\sass folder. Users may change the color, icons, and other layout specific behaviors of the skin with these files. Changes to these files will not have an effect on the content pages if styles have been defined within the source documents or in the Style Designer.
- *Images* - Found in the Pages\images folder. These files are injected into the layout via Connect.asp and can be replaced with another image.

For easy reference, see the below table for a list of template files.

Page Template	Use
Connect.asp	Entry point to the Reverb 2.0 run-time. This file controls the placement of the toolbar buttons, menu (table of contents/index), page, and footer.
Footer.asp	Defines the layout and appearance of the Footer.
Index.asp	Data file for each Reverb parcel's Index information.
Page.asp	Defines the layout and appearance of generated pages.
Parcel.asp	Data file for each Reverb parcel (help set).
Search.asp	Search integration page.
Splash.asp	Defines the layout and appearance of the Reverb 2.0 splash page.
Unsupported_Browser.asp	Defines the layout and appearance of the message that displays when a user visits a Reverb 2.0 help set with an unsupported browser.

Using SASS To Customize WebWorks Reverb 2.0

WebWorks Reverb 2.0 makes use of SASS to produce a responsive HTML5 layout. These files are found in the Pages/sass folder of the Format. SASS file names end with the .scss extension. There are many files in this folder and all of the files have a specific purpose.

Of these files, they fall into two categories:

- *Template Files* - These are identified by their filename. A Template File's name will always begin with the underscore character. These files contain variables that hold values that can be changed by the user to affect areas of the layout without changing the implementation of the layout itself.
- *Layout Files* - These files have filenames that do not begin with the underscore character. These files contain the implementations of the variables defined in template files.

For reference, refer to the below table for a summary of what each SASS file pertains to.

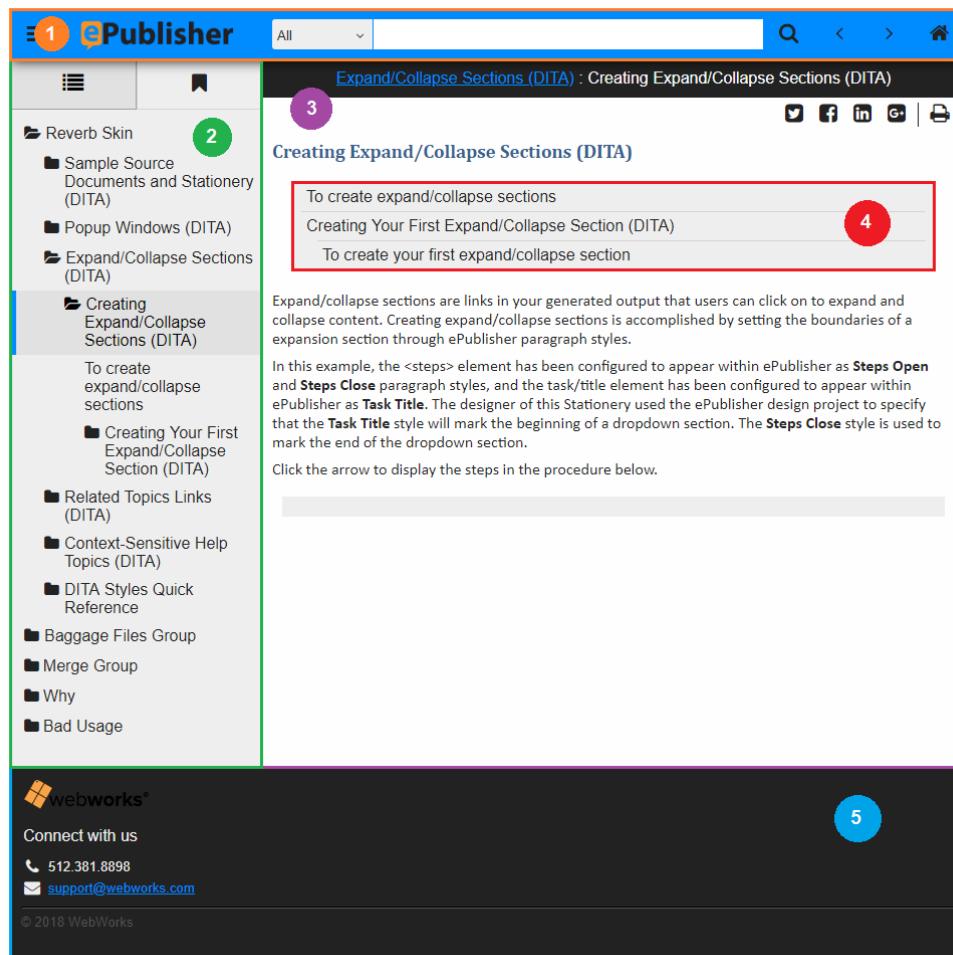
SASS Filename	Pertains To
_borders.scss	Contains variables for border values to be applied across the layout.
_colors.scss	Contains variables for color values to be applied across the layout.
_fonts.scss	Contains variables for font values to be applied across the layout.
_functions.scss	Contains definitions of custom SASS functions created by WebWorks for use in the layout.
_icons.scss	Contains variables for icon values to be applied across the layout.
_sizes.scss	Contains variables for size values to be applied across the layout.
connect.scss	Contains the implementation for top-level structure and core behavior for the layout.
menu_initial_closed.scss	Contains the menu configuration for when the menu is set to be closed upon first visit of the help set.
menu_initial_open.scss	Contains the menu configuration for when the menu is set to be open upon first visit of the help set.
print.scss	Contains style information for the presentation when a page is to be printed.
search.scss	Contains style information for the presentation of the search page.
skin.scss	Contains the aesthetic implementation for the layout. This file is where colors, fonts, borders, icons, and other visual style settings are implemented. This file also contains the various implementations for individual skins.
social.scss	Contains style information for the social buttons in the page toolbar.

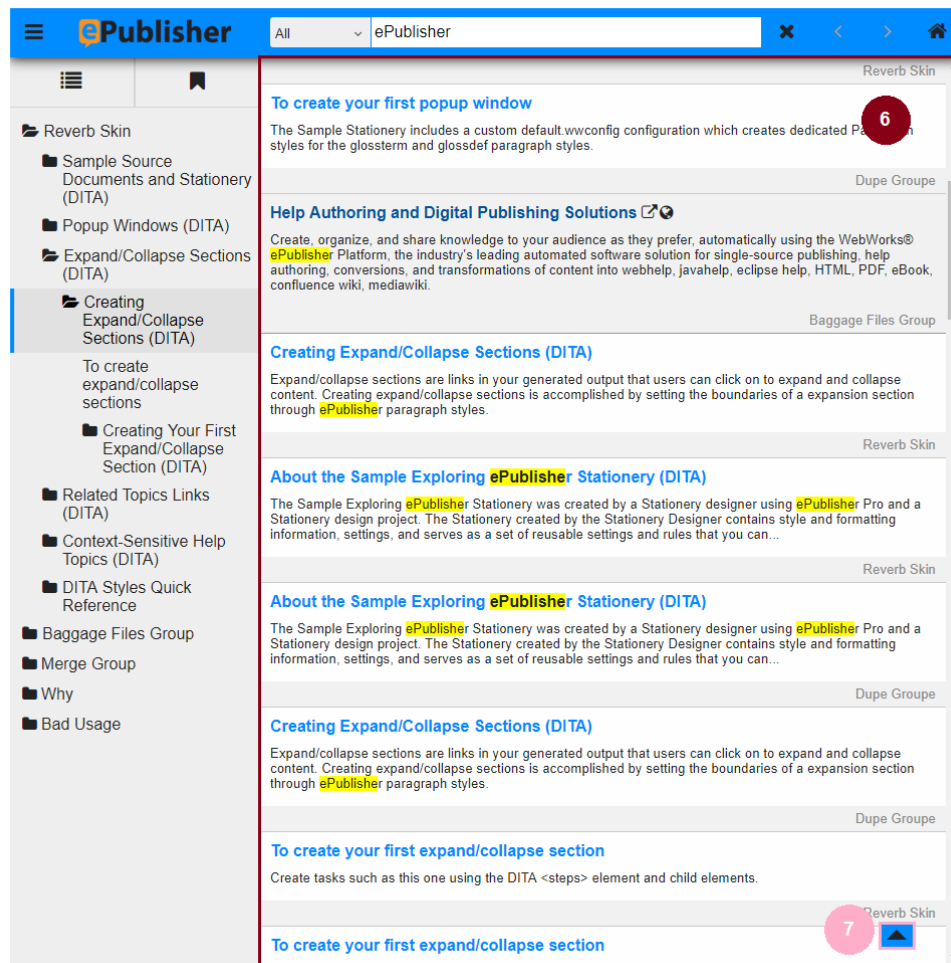
SASS Filename	Pertains To
<code>webworks.scss</code>	Contains default style information for tables, the Mini-TOC, and other various small items in the layout. This file is implemented last and works well for custom implementations as well.

Understanding SASS Variables In WebWorks Reverb 2.0

WebWorks Reverb 2.0 contains an implementation of SASS variables that has been constructed with efficiency and user-friendliness in mind. In each Template File, there are many variables that contain values that are implemented in the layout. Each variable has been given a name that describes the use of the value it holds. Variables are named in a procedural manner, each name starting with the section of the layout that it resides, and with every consecutive word defining more specifically what it pertains to.

As a reference, these graphics provide a visualization of the regions of the layout. The graphics can provide a starting point for finding the variable for the item that needs to be customized. The regions in the graphics have been color coded and numbered from 1 to 7. These numbers can be matched with a variable prefix that matches with a collection of variables that affect that region.





Number	Variable Prefix
1	\$toolbar_*
2	\$menu_*
3	\$page_*
4	\$mini_toc_*
5	\$footer_*
6	\$search_*
7	\$back_to_top_*

Layout Colors In WebWorks Reverb 2.0 Skins

In the file `_colors.scss`, a convention has been created to allow for ease of use and user friendliness. The file makes use of *Layout Colors* that are defined at the top of the page. Every other color value in the layout inherits from these layout colors, so with this logic, you could change the color value of one layout color, and this value would be applied to many aspects of the layout. For the best results, the designer should take a top-down approach, first changing the values of colors at the top level, and then changing individual values for items that need minor adjustments.

For reference, we have created a table for each skin that details every variable that inherits from a certain layout color. These tables may seem like a lot, but just remember, the fact that one Layout Color affects so many values means that for the designer, the workload in the end is greatly reduced.

Neo

Layout Color	Inherited By
<code>\$_layout_color_1</code>	<code>\$back_to_top_background_color,</code> <code>\$footer_link_color,</code> <code>\$link_default_color,</code> <code>\$menu_toc_item_current_highlight_color,</code> <code>\$mini_toc_entry_background_color_hover,</code> <code>\$modal_icon_color,</code> <code>\$page_breadcrumbs_link_color,</code> <code>\$page_breadcrumbs_link_color_hover,</code> <code>\$progress_bar_background_color,</code> <code>\$related_topics_entry_text_color,</code> <code>\$toolbar_background_color,</code> <code>\$toolbar_button_background_color,</code> <code>\$toolbar_button_home_background_color,</code> <code>\$toolbar_button_menu_background_color,</code> <code>\$toolbar_button_next_background_color,</code> <code>\$toolbar_button_previous_background_color,</code> <code>\$toolbar_button_search_background_color,</code> <code>\$toolbar_logo_section_background_color,</code> <code>\$toolbar_search_section_background_color,</code> <code>\$unsupported_browser_heading_text_color</code>
<code>\$_layout_color_2</code>	<code>\$back_to_top_caret_color,</code> <code>\$menu_index_link_color,</code> <code>\$menu_index_text_color,</code> <code>\$menu_index_title_color,</code> <code>\$menu_nav_buttons_icon_color,</code> <code>\$menu_text_color,</code> <code>\$menu_toc_item_icon_color,</code> <code>\$menu_toc_item_text_color,</code> <code>\$menu_toc_title_text_color,</code> <code>\$mini_toc_entry_text_color,</code> <code>\$mini_toc_entry_text_color_visited,</code> <code>\$modal_close_button_color,</code> <code>\$modal_retry_button_background_color,</code> <code>\$modal_text_color,</code> <code>\$page_dropdown_arrow_color,</code> <code>\$page_toolbar_icon_divider_color,</code> <code>\$page_toolbar_social_icon_color,</code> <code>\$page_toolbar_social_icon_facebook_color,</code> <code>\$page_toolbar_social_icon_google_color,</code> <code>\$page_toolbar_social_icon_linkedin_color,</code> <code>\$page_toolbar_social_icon_twitter_color,</code> <code>\$page_toolbar_tool_icon_color,</code> <code>\$related_topics_title_text_color,</code> <code>\$search_filter_message_text_color,</code>

Layout Color	Inherited By
	\$search_plain_text_color, \$search_result_count_message_text_color, \$search_result_icon_color, \$search_result_summary_highlight_text_color, \$search_result_summary_text_color, \$search_title_text_color, \$toolbar_button_home_icon_color, \$toolbar_button_icon_color, \$toolbar_button_menu_icon_color, \$toolbar_button_next_icon_color, \$toolbar_button_previous_icon_color, \$toolbar_button_search_icon_color, \$toolbar_icon_color, \$toolbar_logo_link_text_color, \$toolbar_logo_text_color, \$toolbar_search_scope_option_text_color, \$toolbar_search_scope_options_text_color, \$toolbar_search_scope_selector_icon_color, \$toolbar_search_scope_selector_text_color, \$toolbar_text_color, \$unsupported_browser_message_text_color
\$_layout_color_3	\$disqus_background_color, \$footer_text_color, \$menu_background_color, \$menu_index_background_color, \$menu_nav_buttons_background_color, \$menu_toc_background_color, \$menu_toc_item_background_color, \$mini_toc_background_color, \$toolbar_search_scope_option_background_color, \$toolbar_search_scope_options_background_color, \$toolbar_search_scope_selector_background_color
\$_layout_color_4	\$mini_toc_entry_text_color_hover, \$mini_toc_entry_text_color_visited_hover, \$modal_retry_button_text_color, \$no_javascript_text_color, \$page_breadcrumbs_text_color
\$_layout_color_5	\$footer_background_color, \$page_breadcrumbs_background_color
\$_layout_color_6	\$modal_background_color, \$page_background_color, \$search_background_color, \$search_result_background_color

Classic

Layout Color	Inherited By
\$_layout_color_1	\$footer_link_color, \$link_default_color, \$mini_toc_entry_background_color_hover, \$modal_icon_color, \$progress_bar_background_color, \$related_topics_entry_text_color, \$toolbar_button_home_icon_color_click, \$toolbar_button_home_icon_color_hover, \$toolbar_button_menu_icon_color_click, \$toolbar_button_menu_icon_color_hover,

Layout Color	Inherited By
	\$toolbar_button_next_icon_color_click, \$toolbar_button_next_icon_color_hover, \$toolbar_button_previous_icon_color_click, \$toolbar_button_previous_icon_color_hover, \$toolbar_button_search_icon_color_click, \$toolbar_button_search_icon_color_hover, \$unsupported_browser_heading_text_color
\$_layout_color_2	\$back_to_top_caret_color, \$menu_nav_buttons_icon_color, \$menu_text_color, \$menu_toc_item_icon_color, \$menu_toc_title_text_color, \$mini_toc_entry_text_color, \$mini_toc_entry_text_color_visited, \$modal_close_button_color, \$modal_retry_button_background_color, \$modal_text_color, \$page_dropdown_arrow_color, \$page_toolbar_icon_divider_color, \$page_toolbar_social_icon_color, \$page_toolbar_social_icon_facebook_color, \$page_toolbar_social_icon_google_color, \$page_toolbar_social_icon_linkedin_color, \$page_toolbar_social_icon_twitter_color, \$page_toolbar_tool_icon_color, \$related_topics_title_text_color, \$search_filter_message_text_color, \$search_plain_text_color, \$search_result_count_message_text_color, \$search_result_icon_color, \$search_result_summary_highlight_text_color, \$search_result_summary_text_color, \$search_title_text_color, \$toolbar_button_home_icon_color, \$toolbar_button_icon_color, \$toolbar_button_menu_icon_color, \$toolbar_button_next_icon_color, \$toolbar_button_previous_icon_color, \$toolbar_button_search_icon_color, \$toolbar_icon_color, \$toolbar_logo_link_text_color, \$toolbar_logo_text_color, \$toolbar_search_scope_option_text_color, \$toolbar_search_scope_options_text_color, \$toolbar_search_scope_selector_icon_color, \$toolbar_search_scope_selector_text_color, \$toolbar_text_color, \$unsupported_browser_message_text_color
\$_layout_color_3	\$disqus_background_color, \$footer_text_color, \$menu_background_color, \$menu_index_background_color, \$menu_nav_buttons_background_color, \$menu_toc_background_color, \$menu_toc_item_background_color, \$mini_toc_background_color, \$toolbar_search_scope_option_background_color, \$toolbar_search_scope_options_background_color, \$toolbar_search_scope_selector_background_color

Layout Color	Inherited By
<code>\$_layout_color_4</code>	<code>\$menu_nav_buttons_icon_color_click,</code> <code>\$menu_nav_buttons_icon_color_hover,</code> <code>\$menu_toc_item_current_icon_color,</code> <code>\$menu_toc_item_current_icon_color_click,</code> <code>\$menu_toc_item_current_icon_color_hover,</code> <code>\$menu_toc_item_current_text_color,</code> <code>\$menu_toc_item_current_text_color_click,</code> <code>\$menu_toc_item_current_text_color_hover,</code> <code>\$menu_toc_item_icon_color_click,</code> <code>\$menu_toc_item_icon_color_hover,</code> <code>\$menu_toc_item_text_color_click,</code> <code>\$menu_toc_item_text_color_hover,</code> <code>\$mini_toc_entry_text_color_hover,</code> <code>\$mini_toc_entry_text_color_visited_hover,</code> <code>\$modal_retry_button_text_color,</code> <code>\$no_javascript_text_color,</code> <code>\$page_breadcrumbs_link_color,</code> <code>\$page_breadcrumbs_link_color_hover,</code> <code>\$page_breadcrumbs_link_color_visited,</code> <code>\$page_breadcrumbs_link_color_visited_hover,</code> <code>\$page_breadcrumbs_text_color,</code> <code>\$toolbar_search_scope_option_text_color_hover,</code> <code>\$toolbar_search_scope_selector_icon_color_hover,</code> <code>\$toolbar_search_scope_selector_text_color_hover</code>
<code>\$_layout_color_5</code>	<code>\$footer_background_color,</code> <code>\$footer_copyright_message_text_color,</code> <code>\$footer_hr_color,</code> <code>\$page_breadcrumbs_background_color</code>
<code>\$_layout_color_6</code>	<code>\$modal_background_color,</code> <code>\$page_background_color,</code> <code>\$search_background_color,</code> <code>\$search_result_background_color</code>
<code>\$_layout_color_7</code>	<code>\$back_to_top_background_color,</code> <code>\$back_to_top_background_color_hover,</code> <code>\$menu_toc_item_current_highlight_color,</code> <code>\$toolbar_background_color,</code> <code>\$toolbar_button_background_color,</code> <code>\$toolbar_button_home_background_color,</code> <code>\$toolbar_button_home_background_color_click,</code> <code>\$toolbar_button_home_background_color_hover,</code> <code>\$toolbar_button_menu_background_color,</code> <code>\$toolbar_button_menu_background_color_click,</code> <code>\$toolbar_button_menu_background_color_hover,</code> <code>\$toolbar_button_next_background_color,</code> <code>\$toolbar_button_next_background_color_click,</code> <code>\$toolbar_button_next_background_color_hover,</code> <code>\$toolbar_button_previous_background_color,</code> <code>\$toolbar_button_previous_background_color_click,</code> <code>\$toolbar_button_previous_background_color_hover,</code> <code>\$toolbar_button_search_background_color,</code> <code>\$toolbar_button_search_background_color_click,</code> <code>\$toolbar_button_search_background_color_hover,</code> <code>\$toolbar_logo_section_background_color,</code> <code>\$toolbar_search_section_background_color</code>

Layout Color	Inherited By
<code>\$_layout_color_8</code>	<code>\$menu_toc_item_current_background_color,</code> <code>\$menu_toc_item_current_background_color_click,</code> <code>\$menu_toc_item_current_background_color_hover</code>
<code>\$_layout_color_9</code>	<code>\$menu_nav_buttons_background_color_click,</code> <code>\$menu_nav_buttons_background_color_hover,</code> <code>\$menu_toc_item_background_color_click,</code> <code>\$menu_toc_item_background_color_hover,</code> <code>\$toolbar_search_scope_option_background_color_hover,</code> <code>\$toolbar_search_scope_selector_background_color_hover</code>
<code>\$_layout_color_10</code>	<code>\$menu_index_link_color,</code> <code>\$menu_index_text_color,</code> <code>\$menu_index_title_color,</code> <code>\$menu_toc_item_text_color</code>

Corporate

Layout Color	Inherited By
<code>\$_layout_color_1</code>	<code>\$mini_toc_entry_background_color_hover,</code> <code>\$modal_icon_color,</code> <code>\$page_breadcrumbs_link_color,</code> <code>\$page_breadcrumbs_link_color_hover,</code> <code>\$page_breadcrumbs_link_color_visited,</code> <code>\$page_breadcrumbs_link_color_visited_hover,</code> <code>\$progress_bar_background_color,</code> <code>\$related_topics_entry_text_color,</code> <code>\$unsupported_browser_heading_text_color</code>
<code>\$_layout_color_2</code>	<code>\$back_to_top_caret_color,</code> <code>\$menu_nav_buttons_icon_color,</code> <code>\$menu_nav_buttons_icon_color_click,</code> <code>\$menu_nav_buttons_icon_color_hover,</code> <code>\$menu_text_color,</code> <code>\$modal_close_button_color,</code> <code>\$modal_retry_button_background_color,</code> <code>\$modal_text_color,</code> <code>\$page_toolbar_icon_divider_color,</code> <code>\$page_toolbar_social_icon_color,</code> <code>\$page_toolbar_social_icon_facebook_color,</code> <code>\$page_toolbar_social_icon_google_color,</code> <code>\$page_toolbar_social_icon_linkedin_color,</code> <code>\$page_toolbar_social_icon_twitter_color,</code> <code>\$page_toolbar_tool_icon_color,</code> <code>\$related_topics_title_text_color,</code> <code>\$toolbar_button_home_icon_color,</code> <code>\$toolbar_button_home_icon_color_click,</code> <code>\$toolbar_button_home_icon_color_hover,</code> <code>\$toolbar_button_icon_color,</code> <code>\$toolbar_button_menu_icon_color,</code> <code>\$toolbar_button_menu_icon_color_click,</code> <code>\$toolbar_button_menu_icon_color_hover,</code> <code>\$toolbar_button_next_icon_color,</code> <code>\$toolbar_button_next_icon_color_click,</code> <code>\$toolbar_button_next_icon_color_hover,</code> <code>\$toolbar_button_previous_icon_color,</code> <code>\$toolbar_button_previous_icon_color_click,</code> <code>\$toolbar_button_previous_icon_color_hover,</code> <code>\$toolbar_button_search_icon_color,</code>

Layout Color	Inherited By
	\$toolbar_button_search_icon_color_click, \$toolbar_button_search_icon_color_hover, \$toolbar_icon_color, \$toolbar_logo_link_text_color, \$toolbar_logo_text_color, \$toolbar_text_color
\$_layout_color_3	\$disqus_background_color, \$footer_text_color
\$_layout_color_4	\$menu_index_link_color, \$menu_index_text_color, \$menu_index_title_color, \$menu_toc_title_text_color, \$mini_toc_entry_text_color, \$mini_toc_entry_text_color_visited, \$modal_retry_button_text_color, \$no_javascript_text_color, \$page_breadcrumbs_text_color, \$page_dropdown_arrow_color, \$search_filter_message_text_color, \$search_plain_text_color, \$search_result_count_message_text_color, \$search_result_icon_color, \$search_result_summary_highlight_text_color, \$search_result_summary_text_color, \$search_title_text_color, \$toolbar_search_scope_option_text_color, \$toolbar_search_scope_option_text_color_hover, \$toolbar_search_scope_options_text_color, \$toolbar_search_scope_selector_icon_color, \$toolbar_search_scope_selector_icon_color_hover, \$toolbar_search_scope_selector_text_color, \$toolbar_search_scope_selector_text_color_hover, \$unsupported_browser_message_text_color
\$_layout_color_5	\$toolbar_button_home_background_color_hover, \$toolbar_button_menu_background_color_hover, \$toolbar_button_next_background_color_hover, \$toolbar_button_previous_background_color_hover, \$toolbar_search_section_background_color_hover
\$_layout_color_6	\$menu_background_color, \$menu_index_background_color, \$menu_nav_buttons_background_color, \$menu_toc_background_color, \$menu_toc_item_background_color, \$mini_toc_background_color, \$modal_background_color, \$page_background_color, \$page_breadcrumbs_background_color, \$search_background_color, \$search_result_background_color, \$toolbar_search_scope_option_background_color, \$toolbar_search_scope_options_background_color, \$toolbar_search_scope_selector_background_color
\$_layout_color_7	\$back_to_top_background_color, \$back_to_top_background_color_hover, \$menu_toc_item_current_highlight_color, \$toolbar_background_color,

Layout Color	Inherited By
	\$toolbar_button_background_color, \$toolbar_button_home_background_color, \$toolbar_button_menu_background_color, \$toolbar_button_next_background_color, \$toolbar_button_previous_background_color, \$toolbar_button_search_background_color, \$toolbar_button_search_background_color_click, \$toolbar_button_search_background_color_hover, \$toolbar_logo_section_background_color, \$toolbar_search_section_background_color
\$_layout_color_8	\$toolbar_button_home_background_color_click, \$toolbar_button_menu_background_color_click, \$toolbar_button_next_background_color_click, \$toolbar_button_previous_background_color_click
\$_layout_color_9	\$menu_nav_buttons_background_color_click, \$menu_nav_buttons_background_color_hover
\$_layout_color_10	\$mini_toc_entry_text_color_hover, \$mini_toc_entry_text_color_visited_hover
\$_layout_color_11	\$menu_toc_item_current_icon_color, \$menu_toc_item_current_icon_color_click, \$menu_toc_item_current_icon_color_hover, \$menu_toc_item_current_text_color, \$menu_toc_item_current_text_color_click, \$menu_toc_item_current_text_color_hover, \$menu_toc_item_icon_color, \$menu_toc_item_icon_color_click, \$menu_toc_item_icon_color_hover, \$menu_toc_item_text_color, \$menu_toc_item_text_color_click, \$menu_toc_item_text_color_hover
\$_layout_color_12	\$footer_background_color, \$footer_copyright_message_text_color, \$footer_hr_color

Metro

Layout Color	Inherited By
\$_layout_color_1	\$back_to_top_background_color, \$footer_link_color, \$link_default_color, \$mini_toc_entry_background_color_hover, \$modal_icon_color, \$page_breadcrumbs_link_color, \$page_breadcrumbs_link_color_hover, \$progress_bar_background_color, \$related_topics_entry_text_color, \$unsupported_browser_heading_text_color
\$_layout_color_2	\$back_to_top_caret_color, \$menu_toc_item_current_text_color, \$menu_toc_item_current_text_color_click, \$menu_toc_item_current_text_color_hover, \$menu_toc_item_text_color_click, \$menu_toc_item_text_color_hover, \$mini_toc_entry_text_color,

Layout Color	Inherited By
	\$mini_toc_entry_text_color_visited, \$modal_close_button_color, \$modal_retry_button_background_color, \$modal_text_color, \$page_dropdown_arrow_color, \$page_toolbar_icon_divider_color, \$page_toolbar_social_icon_color, \$page_toolbar_social_icon_facebook_color, \$page_toolbar_social_icon_google_color, \$page_toolbar_social_icon_linkedin_color, \$page_toolbar_social_icon_twitter_color, \$page_toolbar_tool_icon_color, \$related_topics_title_text_color, \$search_filter_message_text_color, \$search_plain_text_color, \$search_result_count_message_text_color, \$search_result_icon_color, \$search_result_summary_highlight_text_color, \$search_result_summary_text_color, \$search_title_text_color, \$toolbar_logo_link_text_color, \$toolbar_logo_text_color, \$toolbar_search_scope_option_text_color, \$toolbar_search_scope_options_text_color, \$toolbar_search_scope_selector_icon_color, \$toolbar_search_scope_selector_text_color, \$toolbar_text_color, \$unsupported_browser_message_text_color
\$_layout_color_3	\$disqus_background_color, \$footer_text_color, \$menu_index_link_color, \$menu_index_text_color, \$menu_index_title_color, \$menu_nav_buttons_background_color_click, \$menu_nav_buttons_icon_color, \$menu_nav_buttons_icon_color_hover, \$menu_text_color, \$menu_toc_item_current_icon_color, \$menu_toc_item_current_icon_color_click, \$menu_toc_item_current_icon_color_hover, \$menu_toc_item_icon_color_click, \$menu_toc_item_icon_color_hover, \$menu_toc_item_text_color, \$menu_toc_title_text_color, \$mini_toc_background_color, \$toolbar_button_home_background_color_click, \$toolbar_button_menu_background_color_click, \$toolbar_button_next_background_color_click, \$toolbar_button_previous_background_color_click, \$toolbar_search_scope_option_background_color, \$toolbar_search_scope_options_background_color, \$toolbar_search_scope_selector_background_color
\$_layout_color_4	\$mini_toc_entry_text_color_hover, \$mini_toc_entry_text_color_visited_hover, \$modal_retry_button_text_color, \$no_javascript_text_color, \$page_breadcrumbs_text_color, \$toolbar_button_home_icon_color, \$toolbar_button_home_icon_color_hover, \$toolbar_button_icon_color, \$toolbar_button_menu_icon_color,

Layout Color	Inherited By
	\$toolbar_button_menu_icon_color_hover, \$toolbar_button_next_icon_color, \$toolbar_button_next_icon_color_hover, \$toolbar_button_previous_icon_color, \$toolbar_button_previous_icon_color_hover, \$toolbar_button_search_icon_color, \$toolbar_button_search_icon_color_hover, \$toolbar_icon_color
\$_layout_color_5	\$footer_background_color, \$menu_toc_item_current_highlight_color, \$page_breadcrumbs_background_color, \$toolbar_background_color
\$_layout_color_6	\$modal_background_color, \$page_background_color, \$search_background_color, \$search_result_background_color
\$_layout_color_7	\$menu_background_color, \$menu_index_background_color, \$menu_nav_buttons_background_color, \$menu_nav_buttons_background_color_hover, \$menu_toc_background_color, \$menu_toc_item_background_color, \$toolbar_button_background_color, \$toolbar_button_home_background_color, \$toolbar_button_home_background_color_hover, \$toolbar_button_menu_background_color, \$toolbar_button_menu_background_color_hover, \$toolbar_button_next_background_color, \$toolbar_button_next_background_color_hover, \$toolbar_button_previous_background_color, \$toolbar_button_previous_background_color_hover, \$toolbar_logo_section_background_color, \$toolbar_search_section_background_color
\$_layout_color_8	\$menu_toc_item_background_color_click, \$menu_toc_item_background_color_hover
\$_layout_color_9	\$menu_toc_item_current_background_color, \$menu_toc_item_current_background_color_click, \$menu_toc_item_current_background_color_hover

Social

Layout Color	Inherited By
\$_layout_color_1	\$back_to_top_background_color, \$footer_link_color, \$link_default_color, \$menu_toc_item_current_highlight_color, \$mini_toc_entry_background_color_hover, \$modal_icon_color, \$page_breadcrumbs_link_color, \$page_breadcrumbs_link_color_hover, \$progress_bar_background_color, \$related_topics_entry_text_color, \$toolbar_background_color, \$toolbar_button_background_color, \$toolbar_button_home_background_color,

Layout Color	Inherited By
	\$toolbar_button_menu_background_color, \$toolbar_button_next_background_color, \$toolbar_button_previous_background_color, \$toolbar_button_search_background_color, \$toolbar_logo_section_background_color, \$toolbar_search_section_background_color, \$unsupported_browser_heading_text_color
\$_layout_color_2	\$back_to_top_caret_color, \$menu_index_link_color, \$menu_index_text_color, \$menu_index_title_color, \$menu_nav_buttons_icon_color, \$menu_text_color, \$menu_toc_item_icon_color, \$menu_toc_item_text_color, \$menu_toc_title_text_color, \$mini_toc_entry_text_color, \$mini_toc_entry_text_color_visited, \$modal_close_button_color, \$modal_retry_button_background_color, \$modal_text_color, \$page_breadcrumbs_text_color, \$page_dropdown_arrow_color, \$page_toolbar_icon_divider_color, \$page_toolbar_social_icon_color, \$page_toolbar_social_icon_facebook_color, \$page_toolbar_social_icon_google_color, \$page_toolbar_social_icon_linkedin_color, \$page_toolbar_social_icon_twitter_color, \$page_toolbar_tool_icon_color, \$related_topics_title_text_color, \$search_filter_message_text_color, \$search_plain_text_color, \$search_result_count_message_text_color, \$search_result_icon_color, \$search_result_summary_highlight_text_color, \$search_result_summary_text_color, \$search_title_text_color, \$toolbar_search_scope_option_text_color, \$toolbar_search_scope_options_text_color, \$toolbar_search_scope_selector_icon_color, \$toolbar_search_scope_selector_text_color, \$unsupported_browser_message_text_color
\$_layout_color_3	\$disqus_background_color, \$footer_text_color, \$menu_background_color, \$menu_index_background_color, \$menu_nav_buttons_background_color, \$menu_toc_background_color, \$menu_toc_item_background_color, \$mini_toc_background_color, \$toolbar_search_scope_option_background_color, \$toolbar_search_scope_options_background_color, \$toolbar_search_scope_selector_background_color
\$_layout_color_4	\$mini_toc_entry_text_color_hover, \$mini_toc_entry_text_color_visited_hover, \$modal_retry_button_text_color, \$no_javascript_text_color, \$toolbar_button_home_icon_color,

Layout Color	Inherited By
	\$toolbar_button_home_icon_color_click, \$toolbar_button_home_icon_color_hover, \$toolbar_button_icon_color, \$toolbar_button_menu_icon_color, \$toolbar_button_menu_icon_color_click, \$toolbar_button_menu_icon_color_hover, \$toolbar_button_next_icon_color, \$toolbar_button_next_icon_color_click, \$toolbar_button_next_icon_color_hover, \$toolbar_button_previous_icon_color, \$toolbar_button_previous_icon_color_click, \$toolbar_button_previous_icon_color_hover, \$toolbar_button_search_icon_color, \$toolbar_button_search_icon_color_click, \$toolbar_button_search_icon_color_hover, \$toolbar_icon_color, \$toolbar_logo_link_text_color, \$toolbar_logo_text_color, \$toolbar_text_color
\$_layout_color_5	\$footer_background_color
\$_layout_color_6	\$modal_background_color, \$page_background_color, \$page_breadcrumbs_background_color, \$search_background_color, \$search_result_background_color
\$_layout_color_7	\$toolbar_button_icon_color_disabled, \$toolbar_button_menu_icon_color_disabled

Format Settings for WebWorks Reverb 2.0

WebWorks Reverb defines several format specific settings.

WebWorks Reverb 2.0 Format Settings

The complete reference of WebWorks Reverb 2.0 Settings.

Format Setting	Use
Browser Tab Icon (favicon)	Specifies a .png .ico or .jpg file to use as the Browser Tab Icon (favicon) for the Reverb 2.0 help set.
Display large images in lightbox	When a thumbnail is used for an image, the full size version of the image can be viewed by clicking the image which is then displayed in a lightbox. When disabled the image displays in a separate file.
Enable Print Icon	Enables/disables the print icon in generated pages
Entry Filename	Specifies the name of the Reverb entry-point file name (default is "index.html")
Feedback Email	Defines the feedback email address for use in generated pages
Feedback Email Message	Defines the contents of the feedback email subject line and body section. Use \$Location; to insert the URL of the current page
Skin	Specifies an alternate plugin file to use for changing the look-and-feel of the Reverb skin.
Use first document as splash page	Determines initial page displayed
WebWorks Help API Compatibility	Enables/disables compatability with the WebWorks Help API allowing Reverb to replace existing implementations of WebWorks Help while using the same API.

WebWorks Reverb 2.0 Toolbar Format Settings

The complete reference of WebWorks Reverb 2.0 Format Settings that affect the toolbar.

Format Setting	Use
Home	Enables/disables the home navigation button which navigates users to the first page in the browse sequence
Linked Toolbar Logo	Enables or Disables linking of the Toolbar Logo. If this setting is Enabled, the user must also set an address in the Toolbar Logo Link Address setting.
Toolbar Logo	Determines what Company Info item should be used, if any. Users can select None , Company name , or Company logo image .
Toolbar Logo Link Address	Determines an address the Linked Toolbar Logo directs to when a user clicks on the Toolbar Logo.
Toolbar Logo Override	Allows the user to set a custom image for the Toolbar Logo. As the name suggests, this setting will override any setting selected in the Toolbar Logo setting.

WebWorks Reverb 2.0 Menu Format Settings

The complete reference of WebWorks Reverb 2.0 Format Settings that affect the menu.

Format Setting	Use
Generate Menu	Determines whether the menu is generated or not.
Menu Initial State	Determines whether the menu is visible or hidden to users when they first visit the help set.
Minimum Page Width for Docked Menu	Sets the minimum width that the viewport must be to allow the menu to use docked behavior. If the viewport is less wide than this value, the user will be presented with the mobile view.

WebWorks Reverb 2.0 Page Format Settings

The complete reference of WebWorks Reverb 2.0 Format Settings that affect the page.

Format Setting	Use
Reverb 2.0 Page Style	Specifies the page style to use when processing the Reverb entry-point file that encapsulates the TOC, Index, Search, and content panels.
Splash Page Style	Specifies the page style to use when processing the splash page (move)

WebWorks Reverb 2.0 Footer Format Settings

The complete reference of WebWorks Reverb 2.0 Format Settings that affect the footer.

Format Setting	Use
Footer Location	Determines where the footer is generated. If End of Layout is selected, the footer will be generated underneath both the menu and the page content. If End of Page is selected, the footer will be generated next to the menu and underneath the page content.
Footer Logo	Determines what Company Info item should be used, if any. Users can select None , Company name , or Company logo image .
Footer Logo Link Address	Determines an address the Linked Footer Logo directs to when a user clicks on the Footer Logo.
Footer Logo Override	Allows the user to set a custom image for the Footer Logo. As the name suggests, this setting will override any setting selected in the Footer Logo setting.
Generate Footer	Determines whether the footer is generated or not.
Linked Footer Logo	Enables or Disables linking of the Footer Logo. If this setting is Enabled, the user must also set an address in the Footer Logo Link Address setting.

Google Format Settings

The complete reference of Google Settings.

Format Setting	Use
Google Analytics Identifier	Enables the use of Google Analytics to measure page activity
Google Translate	Enables support for the Google Translate web service

Social Format Settings

The complete reference of Social Media Settings.

Format Setting	Use
Disqus Identifier	Enables support for user comments using the Disqus comment web service
Disqus - Allow non-public networks	Enable users to post Disqus comments behind a firewall or non-public website
FaceBook Like	Enables users to report “I like this!” in Facebook
Google +1 Button	Enables users to recommend the current page to their Google +1 social networks
LinkedIn Share	Enables users to share the current page to their LinkedIn community as well as increment the counter that shows how many times the link has been shared.
Tweet This!	Enable Twitter “Tweet This!” support

Selecting an Alternate Skin for WebWorks Reverb 2.0

In WebWorks Reverb 2.0, you can select among several alternate skins to define the appearance of your help. When you specify an alternate skin, the underlying ASP and CSS files will be automatically updated to reflect the differences in that skin.

To choose an alternate skin for WebWorks Reverb

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. In the **WebWorks Reverb 2.0** category, select the right column of the **Skin** entry to display the file picker button.
4. Click the file picker button to bring up an **Open** file dialog which will display a list of skin plugin files. Each skin plugin file is identifiable by a `.weplugin` extension.
5. Browse to the plugin file that you wish to use and double-click it to set the skin to that value.

Once you have set an alternate skin, you can later change it again using the same procedure.

Warning: If you have customized any files using the **Advanced** menu to create a Target override, then you will need to remove that customization and then re-implement it again after you change the skin setting. For more information on implementing and managing Target overrides see “Creating Target Overrides”.

Note: If you are going to customize an alternative Reverb 2.0 skin, first set the skin type in the **Target Settings** then create a **Target Customization** override for the file(s) that you want to customize.

Customizing the Top-Level Entry File

Specifying the Entry Page Name

When you generate output for your project, a top-level entry file is created. This file defines the frameset for the help and gives the user a file to open that displays the complete help. By default, the top-level entry file is named `index.html`, but you can specify any name you need for the top-level entry file as long as you specify an `.html` or `.htm` extension.

To rename the top-level file

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. In **Entry filename**, type the name you want to use for the top-level entry file.
4. Click **OK**.
5. Regenerate the project and review the results.

Specifying the Entry Page Style

WebWorks Reverb 2.0 allows users to leverage ePublisher's Style Designer to set color and layout properties of the generated entry page. Users can specify a page style to use under the WebWorks Reverb 2.0 section of the Format Settings dialog.

Customizing the Splash Page in WebWorks Reverb 2.0

You can modify or remove the splash page that is displayed while your WebWorks Reverb 2.0 opens. You can customize the splash page in the Stationery, and then writers can override this customization in each project, as needed.

Specifying the Splash Page Style

WebWorks Reverb 2.0 allows users to leverage ePublisher's Style Designer to set color and layout properties of the generated splash page. Users can specify a page style to use under the WebWorks Reverb 2.0 section of the Format Settings dialog.

Replacing the Splash Image

The splash page is the first page that displays in the topic pane when the help set launches initially. By default, WebWorks Reverb 2.0 systems display the WebWorks Reverb 2.0 splash image. You can replace the default splash image with a custom image.

To replace the splash page image

1. Identify the theme in use for your WebWorks Reverb 2.0 target that you want to modify. For more information, see “Selecting a Theme”.
2. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
3. *If you want to override the image for all WebWorks Reverb 2.0 targets*, create the **Formats**\WebWorks Reverb 2.0\Pages\images folder in your *projectname* folder, where *projectname* is the name of your ePublisher project.
4. *If you want to override the image for one WebWorks Reverb 2.0 target*, create the **Targets**\WebWorks Reverb 2.0\Pages\images folder in your *projectname* folder, where *projectname* is the name of your ePublisher project.
5. Copy the `splash.jpg` file from the following folder to the `images` override folder you created within your project folder:

```
Program Files\WebWorks\ePublisher Designer\Formats\WebWorks Reverb 2.0\ Pages\images
```
6. Open the `splash.jpg` file you copied to your project override folder and modify it to be the splash page image you want.
7. Save and close the `splash.jpg` file.
8. Regenerate your project to review the changes.

Modifying the Splash Page

Users may also create an override for the splash page template, `Splash.asp`. This allows users to change every aspect of a splash page's appearance.

Removing the Splash Page

When WebWorks Reverb 2.0 opens, it displays the splash page. However, instead of displaying the splash page, you can configure WebWorks Reverb 2.0 to display the first topic in the help.

To remove the splash page

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. Set **Use first document as splash page** to **Enabled**.
4. Click **OK**.
5. Regenerate the project and review the results.

Using Context-Sensitive Help in WebWorks Reverb 2.0

Context-sensitive help links provide content based on the context of what the user is doing. In many cases, this help content is based on the window that is open and active. For example, the **Help** button on a window in a software product can open a specific help topic that provides important information about the window and links to related topics.

WebWorks Reverb 2.0 allows you to use a TopicAlias marker to define an internal identifier for each topic. The benefit of using an internal identifier is that it allows file names to change without impacting the links from the product. The writer inserts this marker in a topic and specifies a unique value for that topic. ePublisher creates a mapping file to identify each topic associated with a unique value. Then, WebWorks Reverb 2.0 uses this internal identifier and the mapping file to display the correct topic. Before you can reference topics in WebWorks Reverb 2.0 using topic aliases, you must enable TopicAlias markers in your Stationery. For more information, see “Defining Filename Markers for Context-Sensitive Help Links”.

Understanding Mapping Files in WebWorks Reverb 2.0

WebWorks Reverb 2.0 does not generate a mapping file. However, you can see a list of defined contexts and topics in the parcel file. You can also use the Topics Report to verify that context-sensitive help topics have been created for each topic ID specified in your source document. The Topics Report lists the topic ID and the topic file created for each topic ID. Topic aliases report. For more information about the topics report, see “Understanding Topics Reports”.

Opening Context-Sensitive Help in WebWorks Reverb 2.0 using Standard URLs

You can open WebWorks Reverb 2.0 from the application using standard URLs.

To open a specific topic in WebWorks Reverb 2.0, use the following URL:

```
helplocation/index.html#context/group_context/topic_alias
```

The variable parts of this URL are defined as follows:

helplocation

Specifies the location of the desired WebWorks Reverb 2.0 help set. If the help is on a Web server, specify the location using the `http` protocol and the Web site path to the root of the help, such as `http://www.webworks.com/help`.

group_context

Specifies the group context value for the top-level group in which the topic resides in Document Manager. This group context is specified in merge settings for each top-level group.

topic_alias

Specifies the value of the TopicAlias marker in the topic to open.

URL Commands Support by WebWorks Reverb 2.0

WebWorks Reverb 2.0 supports additional commands in addition to context-sensitive help.

Command	Related Action
index.html# context / <i>group_name</i>	Display specified context-sensitive help topic
index.html# page / <i>child_page_name</i>	Displays define child page
index.html# search / <i>search words</i>	Initiates search for specific terms
index.html# search / <i>search words</i> # scope / <i>group_name</i>	Initiates search for specific terms using only the specified groups
index.html# toc /	Display the table of contents panel
index.html# index /	Display the index panel
index.html# parcels / <i>group_name</i>	Loads the help set with only the specified groups

Opening Context-Sensitive Help in WebWorks Reverb 2.0 using JavaScript

You can use JavaScript to open your context-sensitive help links when working with web applications or websites designed with HTML and JavaScript. For complete details of using WebWorks Reverb with web applications and websites, see [wiki.webworks.com/DevCenter/Projects/Reverb/For Web Applications](http://wiki.webworks.com/DevCenter/Projects/Reverb/For%20Web%20Applications).

Opening Context-Sensitive Help in WebWorks Reverb 2.0 using the WebWorks Help API

In addition to using simple URLs for opening context-sensitive help, you can alternatively use an API that is incorporated into your application. Using an API makes it easier to enable help buttons in your application so that the load the correct page in your Reverb 2.0 output.

WebWorks Reverb 2.0 uses the same API as WebWorks Help 4 and 5. For information about the benefits and steps for using this API, see “Opening Context-Sensitive Help with the WebWorks Help API”.

Steps for Enabling the Context-Sensitive Help API for use with WebWorks Reverb 2.0

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. In the **WebWorks Reverb 2.0** category, set **WebWorksHelpAPICompatibility** to **Enabled**.
4. Click **OK** to save the target settings.
5. Have your application developers download the WebWorks Help SDK located at: wiki.webworks.com/DevCenter/Projects/WebWorksHelp/WebWorksHelpSDK. Depending on the technologies your application uses, your developers will be able to incorporate the API and then make context-sensitive API calls to your published Reverb 2.0 output.

Configuring Client-Side Search for Reverb 2.0

This is the default search configuration for WebWorks Reverb 2.0 in the Target Settings. This search implementation requires no additional setup or configuration. The search implementation supports both wildcard (*) and phrase searching.

Configuring Synonyms

By defining a list of synonyms, you can increase the effectiveness of a search by grouping similar words together as a single result. You can define as synonyms words that are not in your source documents, but the main word does need to be in a source file. For example, if you define `video` and `avi` as synonyms of `movie` (main word), then when a user searches for `video`, a match will be scored showing the source files where the word itself is in, but also where the main word `movie` is as well. Please note that the inverse is not true, meaning that if a user searches instead for `movie`, files where the word `video` is will not be scored because of the synonym entry.

To add these definitions, you'll need to perform an override on a file called `locales.xml`, located by default in the ePublisher Designer installation directory here:

```
\Formats\WebWorks Reverb 2.0\Shared\common\locale
```

Once the file is copied to the correct location in your project directory, open it in any text- or XML-editing application (e.g., NotePad), and add entries to the `<Synonyms>` section. For example, your synonyms might look like this:

```
<Synonyms>
  <Word value="movie">
    <Synonym value="avi" />
    <Synonym value="video" />
  </Word>
  <Word value="designer">
    <Synonym value="pro" />
    <Synonym value="publisher" />
  </Word>
</Synonyms>
```

Please note that the Synonyms feature does not recognize multiple word values or synonyms. And, it ignores words (and synonyms) shorter in length than the specified value for `MinimumWordLength` defined in the same file. Synonyms are not case sensitive.

Searching WebWorks Reverb 2.0 - URL Method

When viewing the URL for your help, you can use a search term after the URL, for example you can go to the link here: <http://www.webworks.com/Documentation/Reverb/index.html#search/ePublisher>

When clicked, the link above searches for the term “ePublisher” this can be changed to any search term, which will produce results accordingly.

Incorporating Google Analytics for Your Reverb 2.0 Files

WebWorks Reverb 2.0 is not only a fully functional help system for your content, it's also a collection of distinct web page files that can be viewed individually on a website. In fact, you can very easily share URLs of individual page files from your Reverb 2.0 help system. These URLs will load just like a single web page within a browser. While this is very nice from an end-user perspective, because the load times are very minimal, it is even better from a publisher perspective. Simply put, you can now easily track the usage patterns and frequencies of all your Reverb 2.0 help pages, which is where the value of Google Analytics comes in; all you have to do is either determine the appropriate Google *Analytics Account ID* that your website is currently using (if Google Analytics has already been setup for your website), or create a Google Analytics Account which will then provide a unique identifier for you website or website path. Since the 2011.1 version of WebWorks Reverb 2.0, you can follow user search patterns with the Google Analytics' Site Search feature. This allows you to review search terms and search start pages for your content. For more information, please refer to the following page:

<http://analytics.blogspot.com/2007/11/site-search-now-available.html>

Once you have obtained the appropriate Google *Analytics Account ID*, then you simply set your ePublisher project's format property called: `Google Analytics Identifier`.

Configuring Commenting and End-User Feedback for Reverb 2.0

WebWorks Reverb 2.0 uses the *Disqus* commenting platform for enabling a very powerful and cost-effective commenting and discussion system. The integration with WebWorks Reverb 2.0 is not only transparent, it is optimized to ensure fast downloads. To enable this feature, all you need to do is set up a Disqus account and then create a *Disqus Site* for your Reverb 2.0 help volume(s). Once you create a *Disqus Site*, you will then use the site's *Disqus Site Shortname* as the setting value for your ePublisher project's format setting called: `Disqus Identifier`. The Disqus Site can then be used to track and manage all the threads used within your deployed Reverb 2.0 help volume(s). For the reports, you can use the Disqus API as offered in the Plus addon: <http://disqus.com/addons>. For more information regarding the API, refer to the documentation: <http://disqus.com/api>

Steps to Create Your First Disqus Site

1. Visit the URL: <http://disqus.com/>. Then select the **Sign Up** button. You will then be taken through a few simple forms which will create your *Disqus Login* as well as your initial *Disqus Site*.
2. In the field `Site URL`, specify the URL that best describes where your Reverb volume(s) will exist on your website.

Note: You can use a single Disqus site for all your Reverb deployments that exist on the same website or you can use separate Disqus sites for each separately deployed Reverb volume. Disqus provides a wide range of solutions for managing and moderating threads so choosing either strategy will be effective and depends more on how your organization's process will be handled.
3. In the field `Site Name`, specify a unique, human readable name that best describes the site. Then check the `Site Shortname` text box field and correct it to suit your preference. The *Site Shortname* will not only be used by ePublisher, but will also become the sub-domain used on Disqus. So if your *Site Shortname* is *epub*, then the URL used for this site's threads will be <http://epub.disqus.com>.
4. On the same page, specify the information for the *Primary Moderator*, other moderators can be added later. Then select the **Continue** button.
5. On the Settings page, specify your language and any optional features that you wish to also use. Then select the **Continue** button. You now have a *Disqus Login* as well as your first *Disqus Site*. No additional steps are required to begin using your new site

Customizations Specific to Microsoft HTML Help

You can customize the appearance and behavior of HTML Help in several ways. For example, you can customize the default size and position of the HTML Help Viewer window. You can also define which buttons are included in the toolbar pane. The following sections describe these HTML Help-specific customizations. For more information about changes you can make to multiple output formats, see “Checklist: Design, Deploy, and Manage Stationery”.

Adjusting the HTML Help Viewer Window Size and Toolbar Buttons

To modify the default HTML Help Viewer window, you need to override the default `template.hhp` file. This file provides the default HTML Help project options and drives the changes specified through the ePublisher options and settings. You can modify various aspects of the default HTML Help Viewer window definition through HTML Help Workshop. For more information about the window definition settings, see the HTML Help Workshop help.

Note: This template file is used to create the `.chm` file for each top-level group in your project. If you have multiple top-level groups, ePublisher applies these settings to the `.chm` file for each top-level group.

To change the default Help Viewer window size and toolbar buttons

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. *If you want to override the default HHP settings for all HTML Help targets in the project*, create the `Formats\Microsoft HTML Help 1.x\Pages` folder in your project folder.
3. *If you want to override the default HHP settings for a specific target*, create the `Targets\targetname\Pages` folder in your project folder, where *targetname* is the name of the target you want to override.
4. Copy the `template.hhp` file from the following folder to the override folder you created within your project folder:

`Program Files\WebWorks\ePublisher Designer\Formats\Microsoft HTML Help 1.x\Pages`
5. In HTML Help Workshop, open the `template.hhp` file you copied to your project override folder.
6. On the **Project** tab, double-click on **Tripane=** under **[Windows]**. HTML Help Workshop displays the Window Types window, which allows you to adjust the Tripane window definition.
7. *If you want to adjust the default window size and position*, complete the following steps:
 - a. On the **Position** tab, click **Autosizer**.
 - b. Adjust the size and position of the sample window to match how you want the default HTML Help Viewer window to open.
 - c. Click **OK** to define the size and position values based on the sample window.
8. *If you want to adjust the buttons included in the toolbar pane*, on the **Buttons** tab, select the buttons you want to include in the toolbar pane. For more information about a field, click the question mark icon, and then click the field for which you want more information.
9. Click **OK**.
10. On the **File** menu, click **Save Project**.
11. On the **File** menu, click **Exit**.
12. Regenerate your project to review the changes.

Creating an Additional HTML Help Window Definition

To create an additional window definition, such as a window without the navigation pane, you need to override the default `template.hhp` file. This file provides the default HTML Help project options. You can add a window definition through HTML Help Workshop, and then use WindowType markers in your source documents to use that window for specific topics in your HTML Help.

Note: This template file is used to create the `.chm` file for each top-level group in your project. If you have multiple top-level groups, ePublisher applies these settings to the `.chm` file for each top-level group.

To create an additional window definition

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. *If you want to override the default HHP settings for all HTML Help targets in the project*, create the `Formats\Microsoft HTML Help 1.x\Pages` folder in your project folder.
3. *If you want to override the default HHP settings for a specific target*, create the `Targets\targetname\Pages` folder in your project folder, where *targetname* is the name of the target you want to override.
4. Copy the `template.hhp` file from the following folder to the override folder you created within your project folder:
`Program Files\WebWorks\ePublisher Designer\Formats\Microsoft HTML Help 1.x\Pages`
5. In HTML Help Workshop, open the `template.hhp` file you copied to your project override folder.
6. On the **Project** tab, double-click on `Tripane=""` under **[Windows]**. HTML Help Workshop displays the Window Types window.
7. Click **Add**.
8. Type the name of the new window, and then click **OK**.
9. Specify all the appropriate values for the window on all the tabs, and then click **OK**. For more information about an option, see the HTML Help Workshop help.
10. On the **File** menu, click **Save Project**.
11. On the **File** menu, click **Exit**.
12. Regenerate your project to review the changes.

Using Context-Sensitive Help in HTML Help

Context-sensitive help links allow you to open a specific help topic. For example, the **Help** button on a window in a software product can open a specific help topic that describes the window and provides links to related topics. The help topic can also be embedded in the window itself, such as an HTML pane that displays the content of the help topic.

You can reference topics in HTML Help using the file name or an internal identifier called a topic ID or topic alias. To use file names, use a Filename marker to assign a file name to a topic. Then, you can open that specific topic with that name. However, if your file naming changes, you need to change the link to the topic. Before you can reference topics in HTML Help using file names, you must enable Filename markers in your Stationery. For more information, see “Defining Filename Markers for Context-Sensitive Help Links”.

To reference topics in HTML Help using an internal identifier, use a TopicAlias marker to define the identifier for each topic. The benefit of using this approach is that it allows file names to change without impacting the links from the product. The writer inserts a TopicAlias marker in a topic and specifies a unique value for that topic. Then, HTML Help uses a mapping file that defines these topic aliases and maps them to the file generated for that topic. Before you can reference topics in HTML Help using topic aliases, you must enable TopicAlias markers in your Stationery. For more information, see “Defining Filename Markers for Context-Sensitive Help Links”.

To simplify the coding of your source documents, you can use the same marker to define both the name and the topic alias for each topic file. In Style Designer, set the **Marker type** option for the marker you want to use to **Filename and topic alias**. However, if you change the value of this marker, you need to change the application that uses this value.

To use context-sensitive help in HTML Help

1. Determine whether you want to manually create and maintain the mapping file, or you want ePublisher to automatically create and maintain the mapping file. For more information, see “Understanding Mapping Files in HTML Help”.
2. In your source documents, use TopicAlias markers to identify the internal identifier for each topic.
3. *If you want to create and maintain the mapping file*, complete the following steps:
 - a. Work with your application developer to determine who will create and maintain the mapping file.
 - b. Store the mapping file in the `Files` folder in your project. Make sure to keep this file up-to-date with the latest version of this file. Both the application and the help must be built using the same version of this file so the help links open the correct topics.
4. Set the mapping options to define your mapping file, such as whether you want ePublisher to generate the mapping file or you want ePublisher to use the mapping file in the `Files` folder in your project. For more information, see “Setting Mapping File Options for HTML Help”.
5. Generate output from your project and test your context-sensitive help links. For more information, see “Testing Context-Sensitive HTML Help”.

Understanding Mapping Files in HTML Help

Implementing context-sensitive help requires cooperation between the help author and the developer of the application that displays the context-sensitive help topics. In a Microsoft Windows application, all user interface controls, such as windows and tabs, have both symbolic and numeric identifiers. These identifiers are also known as topic IDs and map numbers, respectively.

Each topic ID is associated with a map number in a mapping .h file. When an application calls a context-sensitive help topic, it uses the topic ID to display the correct topic. Therefore, both the help and the application must be built with the same mapping file that associates a map number with each topic ID. If the topic IDs and map numbers do not match, the application displays either the wrong topic or no topic.

The mapping .h file lists topic IDs and map numbers. This mapping file is also referred to as a header file. For Microsoft HTML Help, a mapping file is similar to the following sample file:

```
#define IDH_WINDOW_CPRINTERS 1000
```

```
#define IDH_WINDOW_CSERVICES 1010
```

```
#define IDH_WINDOW_CSHARES 1020
```

In this example, `IDH_WINDOW_CPRINTERS` is a topic ID, and `1001` is the corresponding map number. The marker in the appropriate topic has the `IDH_NEW_PROJECT_WINDOW` value. To use the error checking in HTML Help Workshop, all topic alias values start with `IDH_`.

The help is also built with an alias file that maps each topic ID to a filename. The mapping file assigns a number to a topic ID in both the help and the application. The alias file then defines the file associated with that topic ID. ePublisher generates the alias file based on markers inserted into source documents. For Microsoft HTML Help, an alias file is similar to the following sample file:

```
IDH_WINDOW_CPRINTERS=Computer-PrintersTab.htm
```

```
IDH_WINDOW_CSERVICES=Computer-ServicesTab.htm
```

```
IDH_WINDOW_CSHARES=Computer-SharesTab.htm
```

In this example, `IDH_WINDOW_CPRINTERS` is a topic ID and `Computer-PrintersTab.htm` is the path to the .htm file that you want the application to call for context-sensitive help.

When you implement context-sensitive help, you need to work with your application developers to decide how to choose the topic ID for each context-sensitive help topic:

You choose the topic IDs

You can choose a set of topic IDs and embed them in your source documents using TopicAlias markers. When you generate output, ePublisher can generate a mapping and alias file using those topic IDs and assign a unique number to each topic ID. You can provide the generated mapping file to your application developers, who can embed the topic IDs in the application code. You can then maintain this mapping file, or you can allow ePublisher to generate a new file each time you generate the help. Remember to give the updated file to your application developers each time.

Your developers choose the topic IDs

Your application developers can choose a set of topic IDs and embed them in the application code. Then, you can get a copy of the mapping file from your application developers, specify this mapping file in your project settings, and embed the topic IDs in your source documents using TopicAlias markers. In this case, ePublisher does not generate the mapping file, but ePublisher does generate the alias file.

Before you begin to implement context-sensitive help, meet with your application developers to select one of these methods for assigning the topic IDs to use for context-sensitive help links. Once you choose a set of topic IDs, embed them in your source documents using TopicAlias markers and do not change them.

Setting Mapping File Options for HTML Help

In Microsoft HTML Help projects, you can automatically generate a mapping file or use a mapping file supplied by the application developers. If you allow ePublisher to generate the mapping file, make sure you deliver the generated file to your application developers. Both the product and the help must be built with the same version of the mapping file to make sure the context-sensitive help links are correctly mapped.

To set mapping file options

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. *If you want ePublisher to automatically generate a mapping file using your **TopicAlias** markers*, set the **HTML Help custom map file** setting to **none**.
4. *If you want ePublisher to use a mapping file you provide*, set the **HTML Help custom map file** setting to the name of your mapping file. Make sure you store the specified mapping file in the `Files` folder in your project.
5. Click **OK** to save the target settings.

Testing Context-Sensitive HTML Help

The best way to verify that your context-sensitive help topics function correctly is to test the help system with the application that displays the help topics. This testing process ensures the whole implementation works correctly.

However, while developing the help, you can quickly test the topic aliases you have assigned in your source documents to verify that the correct topic is displayed for each topic alias in the help `.chm` file.

Note: If testing a topic alias works in the `.chm` file, but not with a help link in the application, the application may be using the wrong topic alias, or the application may have been built with a different mapping file than the help. For example, as topics are added and removed from the help, the mapping number associated with a topic ID can change in a mapping file generated by ePublisher.

To test context-sensitive HTML Help

1. Open the `.hhp` file in HTML Help Workshop.
2. On the **Test** menu, click **HTMLHelp API**.
3. In **Compiled file**, select the `.chm` file.
4. In **Command**, select **HH_HELP_CONTEXT**.
5. In **Map number**, specify the map number for the associated topic ID from the `.h` mapping file.
6. Click **Test**.

Defining What's This (Field-Level) Help in HTML Help

The Microsoft HTML Help output format allows you to create What's This field-level help topics. What's This style topics are small, popup windows that provide information about an individual field on a window. To display this type of help for a field, the user clicks a question mark icon next to the minimize, maximize, and close buttons in the title bar of the window, and then clicks on the field.

Note: Due to restrictions in Microsoft HTML Help, you can include only unformatted text in a What's This style topic. You cannot control the font or formatting, and you cannot include tables or images.

To enable What's This style help

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. On the **Project** menu, click a target in the **Active Target** menu option that is the Microsoft HTML Help output format.
4. In **Paragraph Styles**, select the paragraph style you want to use for What's This help content, such as **WhatIsThis**.
5. On the **Options** tab, set **What is this marker** to **Define** or **Define with no output**.
6. In **Marker Styles**, select the marker style you want to use to define the topic ID for the What's This help content, such as **WhatIsThisID**.
7. On the **Options** tab, set **Marker type** to **What is this ID**.

Before the writers begin to implement What's This style help, they need to meet with their application developers to define the topic IDs to use for each field on each window. Once they choose a set of topic IDs, they need to embed them in your source documents using **WhatIsThisID** markers and do not change them.

Writers include the content paragraphs in their source documents and apply the paragraph style you defined. The writers include a **WhatIsThisID** marker in each **WhatIsThis** paragraph that contains the topic ID for that paragraph. ePublisher generates the **WhatIsThis.h** mapping file with the help. You can provide the generated mapping file to your application developers, who can embed the topic IDs in the application code and build the application with the generated file. ePublisher generates an updated file each time you generate the help. Remember to give the updated file to your application developers each time.

Customizations Specific to Simple HTML

You can customize the appearance of Simple HTML in several ways. For example, you can customize the default text, background, and link colors. The following sections describe these Simple HTML-specific customizations. For more information about changes you can make to multiple output formats, see “Checklist: Design, Deploy, and Manage Stationery”.

Removing the Left and Right Margins from Simple HTML Pages

By default, pages in the Simple HTML output format that have a header or footer include blank space to the left and right of all body content. To make more efficient use of the space available in the browser window, you may want to remove the left and right margins. You can do this by overriding the `Page.asp` template file.

To remove the left and right margins

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. *If you want to override the default settings for all Simple HTML targets in the project*, create the `Formats\Simple HTML\Pages` folder in your project folder.
3. *If you want to override the default settings for a specific target*, create the `Targets\targetname\Pages` folder in your project folder, where `targetname` is the name of the target you want to override.
4. Copy the `Page.asp` file from the following folder to the override folder you created within your project folder:

Program Files\WebWorks\ePublisher Designer\Formats\Simple HTML\Pages

5. Open the `Page.asp` file you copied to your project override folder.
6. Find the following block of code and change `blockquote` to `div` in both places:

```
<blockquote wwpage:condition="header-footer-exists" wwpage:content="content">  
Page content with header/footer.  
</blockquote>
```

7. Save and close the `Page.asp` file.

Customizations Specific to Dynamic HTML

You can customize the appearance of Dynamic HTML in several ways. For example, you can customize the appearance of the table of contents and the index entries. The following sections describe these Dynamic HTML-specific customizations. For more information about changes you can make to multiple output formats, see “Checklist: Design, Deploy, and Manage Stationery”.

Using SASS to change the Appearance of Dynamic HTML

To facilitate more advanced CSS handling you can enable a Target Setting in the group **Files** called **Use SASS to compile CSS**. If this setting is **Enabled** `webworks.css` will be generated from the SASS file called `webworks.scss`. This file is located in the `css` folder next to the `webworks.css` file.

Modifying the Appearance of the Table of Contents in Dynamic HTML

ePublisher stores CSS settings that control the appearance of table of contents entries in the `webworks.css` file. You can create an override file to modify these settings for specific levels of the table of contents. For example, you can define a different font size and margin for each level in the table of contents.

To modify the appearance of the table of contents

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. *If you want to override the CSS settings for all Dynamic HTML targets in the project*, create the `Formats\Dynamic HTML\Pages\css` folder in your project folder.
3. *If you want to override the CSS settings for a specific target*, create the `Targets\targetname\Pages\css` folder in your project folder, where `targetname` is the name of the target you want to override.
4. Copy the `webworks.css` file from the following folder to the override folder you created within your project folder:
`Program Files\WebWorks\ePublisher Designer\Formats\Dynamic HTML\Pages\css`
5. Open the `webworks.css` file you copied to your project override folder.
6. Find the code for `div.WebWorks_TOC_Levelx`, where `x` is the level number you want to modify. Then, specify the values within the braces to modify the font or margin:
 - To modify the font of all table of contents entries for the specified level, specify the name of the font you want, such as `font-family: Arial;`.
 - To modify the font size of all table of contents entries for the specified level, specify the size of the font you want, such as `font-size: 14pt;`.
 - To modify the left margin indent of all table of contents entries for the specified level, specify the indent you want, such as `margin-left: 10px;`.
7. Save the `webworks.css` file.
8. Regenerate your project to review the changes.

For example, the following figure illustrates how you could customize your table of contents entries.

```
div.WebWorks_TOC_Level1
```

```
{ font-size: 14pt;
```

```
font-family: Arial;
```

```
margin-left: 12px;
```

```
}
```

```
div.WebWorks_TOC_Level2
```

```
{ font-size: 12pt;
```

```
font-family: Arial;
```

```
margin-left: 24px;
```

```
}
```

Modifying the Appearance of the Index in Dynamic HTML

ePublisher stores CSS settings that control the appearance of index entries in the `webworks.css` file. You can create an override file to modify these settings for specific index entry levels. For example, you can define a different font size and margin for each level in the index.

To modify the appearance of the index

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. *If you want to override the CSS settings for all Dynamic HTML targets in the project*, create the `Formats\Dynamic HTML\Pages\css` folder in your project folder.
3. *If you want to override the CSS settings for a specific target*, create the `Targets\targetname\Pages\css` folder in your project folder, where `targetname` is the name of the target you want to override.
4. Copy the `webworks.css` file from the following folder to the override folder you created within your project folder:
`Program Files\WebWorks\ePublisher Designer\Formats\Dynamic HTML\Pages\css`
5. Open the `webworks.css` file you copied to your project override folder.
6. Find the code for `div.WebWorks_Index_Levelx`, where `x` is the level number you want to modify. Then, specify the values within the braces to modify the font or margin:
 - To modify the font of all index entries for the specified level, specify the name of the font you want, such as `font-family: Arial;`.
 - To modify the font size of all index entries for the specified level, specify the size of the font you want, such as `font-size: 14pt;`.
 - To modify the left margin indent of all index entries for the specified level, specify the indent you want, such as `margin-left: 10px;`.
7. Save the `webworks.css` file.
8. Regenerate your project to review the changes.

For example, the following figure illustrates how you could customize your index entries.

```
div.WebWorks_Index_Level1
```

```
{ font-size: 14pt;
```

```
font-family: Arial;
```

```
margin-left: 12px;
```

```
}
```

```
div.WebWorks_Index_Level2
```

```
{ font-size: 12pt;
```

```
font-family: Arial;
```

```
margin-left: 24px;
```

```
}
```

Other Changes to Text in the TOC and Index in Dynamic HTML

You can change the text in the table of contents and the index by adding the proper CSS coding between the braces for the appropriate style and class. Put the value after a colon, and put a semicolon at the end of the added coding. The following table summarizes the CSS coding for some common modifications.

Markup	Possible Values	Explanation
font-style:	normal italic oblique	Specifies whether the font should use the normal, also known as upright or roman, italic, or oblique faces within a font family.
font-weight:	normal bold bolder lighter 100 200 300 400 500 600 700 800 900	Specifies the thickness of the font. The value <code>normal</code> is equal to 400, and the value <code>bold</code> is equal to 700.
text-transform:	capitalize uppercase lowercase none	Specifies whether to transform the case of the text.
text-align:	left right center justify	Specifies how to align the text on the page.

Customizations Specific to Oracle Help and Sun JavaHelp

You can customize the appearance and behavior of Oracle Help and Sun JavaHelp in several ways. For example, you can customize which buttons are included in the toolbar pane. The following sections describe these Oracle Help and Sun JavaHelp-specific customizations. For more information about changes you can make to multiple output formats, see “Checklist: Design, Deploy, and Manage Stationery”.

Defining the Navigation Pane in Oracle Help

You can specify whether to include the Search tab in your Oracle Help output.

To specify whether to include the Search tab in your Oracle Help

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. In the **Oracle Help** category, set **Enable Search tab** to **Enabled** or **Disabled**.
4. Click **OK** to save the target settings.

Using Custom Windows in Oracle Help

By default, Oracle Help uses the standard Oracle Help viewer. You can modify the size, position, and other characteristics of the Oracle Help windows. You can also define and use custom windows in your Oracle Help project. To define custom windows, you need to override the `template.hs` file, which is used to create your `helpset.hs` file.

Once you have defined custom windows in a project, you can assign topics to them by adding the `WindowType` marker in specific topics in your source documents. When you assign a topic to a custom window, the topic is displayed in that window whenever users view the topic.

To override the `template.hs` file in Oracle Help

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Understanding Stationery, Projects, and Overrides”.
2. *If you want to override the default settings for all Oracle Help targets in the project*, create the `Formats\Oracle Help\Pages` folder in your project folder.
3. *If you want to override the default settings for a specific target*, create the `Targets\targetname\Pages` folder in your project folder, where `targetname` is the name of the target you want to override.
4. Copy the `template.hs` file from the following folder to the override folder you created within your project folder:

`Program Files\WebWorks\ePublisher Designer\Formats\Oracle Help\Pages`
5. Open the `template.hs` file you copied to your project override folder.
6. Modify the `template.hs` file as needed. For more information about this file, see the Oracle Help documentation.
7. Save and close the `template.hs` file.
8. Regenerate your output and review the results.

Defining the Navigation Pane in Sun JavaHelp

By default, the navigation pane includes the Contents, Index, Search, and Favorites tabs. You can select whether to include the Favorites, Glossary, and Search tabs in your output. You can also define your own views in Sun JavaHelp by overriding the `template.hs` file. For example, you can create an alternate list of topics and provide that list as a separate view.

To specify which tabs to include in your Sun JavaHelp

1. On the **Project** menu, select the **Active Target** you want to specify settings for.
2. On the **Target** menu, click **Target Settings**.
3. In the **Java Help** category, set each tab you want to include to **Enabled**.
4. Click **OK** to save the target settings.

Using Context-Sensitive Help in Oracle Help and Sun JavaHelp

Context-sensitive help links allow you to open a specific help topic. For example, the **Help** button on a window in a software product can open a specific help topic that describes the window and provides links to related topics.

You can reference topics in Oracle Help and Sun JavaHelp using the file name or an internal identifier called a topic ID or topic alias. To use file names, use a Filename marker to assign a file name to a topic. Then, you can open that specific topic with that file name. However, if your file naming changes, you need to change the link to the topic. To use an internal identifier, use a TopicAlias marker to define the identifier for each topic. The benefit of using this approach is that it allows file names to change without impacting the links from the product. The writer inserts this marker in a topic and specifies a unique value for that topic. Then, Oracle Help and Sun JavaHelp use a mapping file that defines these topic aliases.

To simplify the coding of your source documents, you can use the same marker to define both the name and the topic alias for each topic file. In Style Designer, set the **Marker type** option for the marker you want to use to **Filename and topic alias**. However, if you change the value of this marker, you need to change the application that uses this value.

To use context-sensitive help in Oracle Help and Sun JavaHelp

1. Meet with your application developers and define the topic ID for each context-sensitive help topic. Also discuss how ePublisher generates the mapping file. For more information, see “Understanding Mapping Files in HTML Help”.
2. In your source documents, use TopicAlias markers to identify the topic ID for each topic.
3. Generate output from your project and test your context-sensitive help links. For more information, see “Testing Context-Sensitive HTML Help”.

Understanding Mapping Files in Oracle Help and Sun JavaHelp

Implementing context-sensitive help requires cooperation between the help author and the developer of the application that displays the context-sensitive help topics. You both need a mapping .jhm file that associates topic IDs with the target URL. When an application calls a context-sensitive help topic, it uses the topic ID to display the correct topic. Therefore, both the help and the application must use the same mapping file. If the topic IDs and target URLs do not match, the application displays either the wrong topic or no topic.

The mapping .jhm file lists topic IDs and target URLs. This mapping file is also referred to as a header file. For Oracle Help and Sun JavaHelp, the mapping file is similar to the following sample file:

```
<mapID target="ch1_htm_999374" url="ch1.htm#999374">
```

```
<mapID target="ch2_htm_999640" url="ch2.htm#999640">
```

```
<mapID target="ch9_htm_999786" url="ch9.htm#999786">
```

In this example, `ch1_htm_99374` is a topic ID, and `ch1.htm#99374` is the target URL for this particular topic ID. The marker in the appropriate topic has the `ch1_htm_999374` value.

When you implement context-sensitive help, you need to work with your application developers to decide how to choose the topic ID for each context-sensitive help topic. You can choose a set of topic IDs and embed them in your source documents using TopicAlias markers. When you generate output, ePublisher generates a mapping file using those topic IDs and assigns the target URL to each topic ID based on your source documents. You can provide the generated mapping file to your application developers, who can embed the topic IDs in the application code. ePublisher generates an updated file each time you generate the help. Remember to give the updated file to your application developers each time.

Note: Once you choose a set of topic IDs, embed them in your source documents using TopicAlias markers and do not change them.

Testing Context-Sensitive Oracle Help and Sun JavaHelp

The best way to verify that your context-sensitive help topics function correctly is to test the help system with the application that displays the help topics. This testing process ensures the whole implementation works correctly.

Customizations Specific to Eclipse Help

You can customize the appearance and behavior of Eclipse Help in several ways. For example, you can specify context plug-in IDs for your Eclipse Help system. You can also specify topic descriptions for context-sensitive help topics.

Using Markers to Specify Context Plug-ins in Eclipse Help

You can specify Eclipse Help context plug-ins by using Context Plugin markers in your source documents. Obtain the context plug-in IDs you need to specify for your source document groups from your development team. ePublisher places the context plug-ins you specify in your source documents in the `plugin.xml` file generated for each source document group you have in your project. The Eclipse developers use the context plug-ins defined in `plugin.xml` files to call your Eclipse Help system as appropriate from Eclipse plug-ins.

To enable specifying context plug-ins for Eclipse Help systems, you need to enable the Context Plugin marker. By default, ePublisher sets the **Marker type** option for a marker named Context Plugin to **Context Plugin**. You can create a marker with a different name and set the **Marker type** option for that marker to **Context Plugin**.

Then, writers can use this marker in the source documents to define context plug-in IDs for each of the source document groups in their project. Context plug-in IDs must follow these guidelines:

- Must be unique
- May specify only one context plug-in ID in each Context Plugin marker
- May contain alphanumeric characters
- Should not contain special characters or spaces, with the exception of underscore (`_`) characters

To assign context plug-in behavior to context plug-in markers

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In the **Marker Styles**, select the marker style you want to modify.
4. On the **Options** tab, set **Marker type** to **Context Plugins**.

Using Markers to Specify Topic Descriptions for Context-Sensitive Help Topics in Eclipse Help

In Eclipse Help, you can specify the topic description you want to display for each context-sensitive link. When you use a `TopicAlias` marker to create context-sensitive links, Eclipse creates a `contexts.xml` file that lists all of the context IDs for the Eclipse Help system you created using `TopicAlias` markers. In the `context.xml` file, Eclipse also provides a description of the context-sensitive link. By default, the description Eclipse provides for the context-sensitive link is the text of the first paragraph of the topic. However, if you want to specify a different description for the context-sensitive link, you can do this by using the `TopicDescription` marker.

To enable specifying topic descriptions for context-sensitive help topics in Eclipse Help, you need to enable the `TopicDescription` marker. By default, ePublisher sets the **Marker type** option for a marker named `TopicDescription` to **TopicDescription**. You can create a marker with a different name and set the **Marker type** option for that marker to **TopicDescription**. Then, writers can use this marker in the source documents to specify topic descriptions for each of context-sensitive help topics in their project.

To assign topic description behavior to topic description markers

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In the **Marker Styles**, select the marker style you want to modify.
4. On the **Options** tab, set **Marker type** to **TopicDescription**.

Customizations Specific to Wiki Markup

You can customize the appearance and behavior of Wiki Markup in several ways. For example, you can specify Wiki categories, or labels, for Wiki topics. You can also specify a table file naming pattern for ePublisher to use when generating Wiki - MoinMoin output.

Defining Wiki Categories or Labels

On Wikis, **categories**, which are referred to as **labels** in some Wiki formats, are used to organize Wiki content. Categories help group together pages that have similar subjects.

Note: MoinMoin and Media Wiki use the term *category* to describe page grouping functionality. Confluence uses the term *label* to describe page grouping functionality.

Pages are assigned to category groups through the use of category or label tags. When you assign a category or label tag to a Wiki page, the category to which the Wiki page belongs displays in a box at the bottom of the page.

Category and label tags on Wiki pages allow categorized pages to automatically be added to a list on a category page on the Wiki. The category page lists all of the Wiki pages tagged for a certain category. For example, if you tag each page on a Wiki that contains licensing information with a Licensing category tag, then a licensing category page on the Wiki can display a list of all of the pages tagged as containing licensing information.

If you enable Wiki category functionality in your Stationery, writers can use the WikiCategory marker in their source documents to assign topics to a category. The category writers assign to a topic will display in a box at the bottom of the Wiki page when they generate Wiki output, and the topic can be listed as a link on a category page on the Wiki.

To enable specifying category or label tags in source documents you need to enable the WikiCategory marker. By default, ePublisher sets the **Marker type** option for a marker named WikiCategory to **WikiCategory**. You can create a marker with a different name and set the **Marker type** option for that marker to **WikiCategory**. Then, writers can use this marker in the source documents to specify categories for Wiki topics. For more information about which output formats support this feature, see “Features Available in Each Output Format”.

To assign WikiCategory behavior to WikiCategory markers

1. Open your Stationery design project.
2. On the **View** menu, click **Style Designer**.
3. In the **Marker Styles**, select the marker style you want to modify.
4. On the **Options** tab, set **Marker type** to **WikiCategory**.

Defining Table File Naming Patterns for Wiki - MoinMoin Output

When ePublisher generates Wiki - MoinMoin output, it creates separate files for each table in a topic. You can specify the table file naming pattern you want ePublisher to use for tables when generating Wiki - MoinMoin output.

Understanding Default File Naming for Tables in Wiki - MoinMoin Output

ePublisher assigns a default file name for each table in a topic when generating Wiki - MoinMoin output by applying the following default naming rule:

SourceDocumentName.SourceDocumentNumber.TopicNumber.TableNumber

The parts of the default naming rule are defined as follows:

SourceDocumentName

Identifies the name of the source document that the topic came from without the file extension.

SourceDocumentNumber

Identifies the number of the source document in the order it is included in its containing group in the project, such as 1 for the first source document in a group and 2 for the second source document in a group. This value starts at 1 for the first source document in each group in your project.

TopicNumber

Identifies the number of the topic (output page) generated from the source document, such as 1 for the first topic generated from a source document and 2 for the second topic generated from a source document. This value starts at 1 for the first topic in each source document.

TableNumber

Identifies the number of the table generated for the topic, such as 1 for the first table generated for a topic in a source document and 2 for the second table generated for a topic in a source document. This value starts at 1 for the first table in a topic.

For example, if you have a `MyFile.fm` source document that contains a topic with three tables, and the topic starts with a paragraph style that has a page break priority set in Style Designer, ePublisher generates tables with the following default table file names when generating Wiki - MoinMoin output: `MyFile.1.1.1`, `MyFile.1.1.2`, and `MyFile.1.1.3`.

Specifying Table File Naming Patterns for Wiki - MoinMoin Ouput

You can specify the table file naming pattern that you want ePublisher to use when generating Wiki - MoinMoin output. For example, you can specify if you would like to include the following items in table file names when you generate Wiki - MoinMoin output:

- Target name
- Name of the group in Document Manager that contains the topic with the table
- Page heading text or title of the topic that contains the table

To specify a table file naming pattern for a target that uses the MoinMoin Wiki output format

1. On the **Project** menu, select a target next to **Active Target** that uses the Wiki - MoinMoin output format.
2. On the **Target** menu, click **Target Settings**.
3. Under **Files**, specify the appropriate values for the table file naming pattern you want to use. For more information about the table naming pattern setting and values, click **Help**.
4. Click **OK**.

7

Designing Wiki Output

With ePublisher, you can quickly and easily populate a Wiki with content from Adobe FrameMaker, Microsoft Word, and DITA-XML source documents. A **Wiki** is a website that uses Wiki software, such as Confluence, MediaWiki, or MoinMoin, to create a web-based collaborative authoring environment. One of the best-known Wikis is the collaborative encyclopedia Wikipedia. Wikis allow users with appropriate permissions to quickly and easily create and edit any number of interlinked Web pages using a simplified markup language or a text editor within a standard Web browser. The Web-based nature of Wikis eliminates the barrier of requiring users to install special software before they can view, create, or edit content.

Wikis are used in many corporations to host corporate and departmental Intranets, project communication web pages, and technical documentation web pages. For example, software development teams who use Agile development processes often use Wikis to quickly create, share, and update project and product information as they work towards the release of a new product or update an existing product.

Why Use a Wiki to Deliver Content?

In today's fast-paced information-based world, content is expensive to develop and maintain, and information about products, services, and processes changes rapidly. More and more, some companies are looking for ways not only to produce traditional content deliverables such as online help systems and PDF files, but also to produce content in a format that can be quickly and easily maintained in real time after it is released. Some companies also want the ability for different groups, such as customers, technical support, and technical sales, that use different content authoring tools, to contribute to efforts that maintain, enhance, and extend product documentation using a simple, universal content authoring tool.

For example, content published in an online help system or in a PDF file may describe how to perform a process at the time a software product releases. However, a few weeks or months later, a user in technical support, a technical sales representative, or a customer may discover an error in the information or identify new information that should accompany the procedure.

In a traditional online help or PDF content delivery model, the change to the procedure would likely not be incorporated into the published documentation set until the next scheduled product release. However, if the content has been published on a Wiki, a user with appropriate permissions can access the page on the Wiki that contains the content using a standard Web browser, quickly edit the page to include other new information about the procedure, and then publish the updated information to users immediately.

ePublisher is the only tool available today that allows content authors to create content with their preferred content authoring tools and deliver the same content in online help systems and PDFs directly to a Wiki. With ePublisher, content that resides in many different content authoring tools and with many different content authors can quickly be generated and deployed to one or more Wikis and shared with users. Users can then use Wiki functionality to comment on the content and edit, enhance, and extend the content.

For example, documentation teams that author in DITA-XML, training teams that author in Adobe FrameMaker, and technical support and product management teams that author in Microsoft Word can all use ePublisher to take their existing source documents and generate and deploy content to a Wiki.

Wiki technology provides the following benefits to content authors:

- Content authors, including technical writers, customers, and other technical users, such as users in technical support, product management, or post-sales services, can update incorrect information or add new information just-in-time.
- Content authors can easily see updates and additions other content authors and users have made to documentation.
- Content authors can use the edits and updates made on Wiki pages to identify changes that should be incorporated back into their source documents and included in future releases.

Wiki technology provides the following benefits to users:

- Viewing content on Wiki pages is easy. Users use a standard Web browser to view Wiki pages.
- Searching and navigating content on a Wiki is intuitive to most users.
 - Wiki content can direct users to additional content through the use of hyperlinks.
 - Some Wikis track the pages users visit and display an interactive “breadcrumb” trail so users can easily return to previously viewed pages as needed.

- Users can use a familiar Google-like search and then click on links in the search results to access the page they want.
- Adding comments and editing content on a Wiki is easy.
 - Most Wikis allow users to add comments and edit content by simply clicking on a button and then typing in their comments and edits much like in a regular word processor.
 - Several Wikis offer What You See Is What You Get (WYSIWYG) formatting and editing controls similar to the controls provided in popular word processing applications. If users do need to use specific Wiki markup language, most are easy to learn, and users can master the basics in minutes.

Deploying your content to a Wiki may be appropriate for you under the following conditions:

- Your content changes frequently
- Your content authors often learn critical information after content has been published
- You do not have enough time to document every concept, procedure, option, and use case before you have to publish your content
- You have groups of internal or external users who can contribute additional information that can improve, enhance, or extend the content
- You have user communities with practioners (users who use the product in the real world every day) who can contribute their knowledge and share best practices and real-world usage scenarios
- Your users are used to searching the web to obtain the most recent information about a product or procedure
- You need to reduce content production and deployment bottlenecks and get new and updated content to users faster
- You want to reduce or remove departmental boundaries around who can view, edit, and update content
- You want a centralized point for collecting changes to content
- You want to link the version of your documentation that you ship with a product to a more recent version maintained online and updated in real time

Checklist: Preparing to Deliver Content on a Wiki

If you want to use ePublisher to generate Wiki output and deploy the output to a Wiki server, use the following checklist to help you plan your Wiki content delivery.

<input checked="" type="checkbox"/>	Task
<input type="checkbox"/>	1. Identify who will use the content you publish to the Wiki. For more information, see "Identifying Wiki Content Users".
<input type="checkbox"/>	2. Identify your Wiki server and Wiki output format. For more information, see "Identifying Your Wiki Server and Wiki Output Format".
<input type="checkbox"/>	3. Define how content will be maintained on the Wiki. For more information, see "Defining the Wiki Content Maintenance Process".
<input type="checkbox"/>	4. Define the navigation you want to provide on your deployed Wiki pages. For more information, see "Defining Wiki Navigation".
<input type="checkbox"/>	5. Ensure you understand the search capabilities provided by your Wiki server. For more information, see "Understanding Wiki Search Capabilities".
<input type="checkbox"/>	6. Create landing pages for your Wiki output on the Wiki server. For more information, see "Creating Landing Pages".
<input type="checkbox"/>	7. Define and configure permissions for your users on the Wiki server. For more information, see "Defining and Configuring Wiki Permissions".
<input type="checkbox"/>	8. Verify your Wiki server is configured correctly. For more information, see "Verifying Your Wiki Server Configuration".
<input type="checkbox"/>	9. Generate and deploy your Wiki content. For more information, see "Generating and Deploying Wiki Content".

Identifying Wiki Content Users

Identify who will view, comment on, edit, and add to the content you deploy to a Wiki. You may want to allow some of the following users to view your Wiki content:

- Customers
- Partners
- Technical support
- Sales
- Documentation
- Engineering
- Product management
- Marketing
- Human resources

Also identify who can add comments about the content, edit the content, and add new content pages. For example, if your documentation team will deploy the product installation, configuration, and usage content they develop to a Wiki, who should be able to view the content and add comments and edits to the deployed Wiki pages? Should everyone in the company be able to view the content, but only documentation writers and editors be able to edit and update the Wiki pages? Or should others in the company, such as technical support, quality assurance, and product management, also be able to add comments, edit content, and add additional pages?

Will you allow some or all of your customers and partners to view and edit the content on the deployed Wiki pages and add comments and new pages?

The following table can help you identify who should be able to view content, add comments about content, and edit content on your deployed Wiki pages. Also identify if you want to allow users to be able to add new pages to the Wiki or delete pages.

User/Role	View Content	Add Comments	Edit Content	Add New Pages	Delete Pages
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Identifying Your Wiki Server and Wiki Output Format

Before you begin generating Wiki output, first identify an available Wiki server to which you can deploy your Wiki content. The type of Wiki server you use will determine the Wiki format you will use when generating content with ePublisher. For example, if you plan to deploy content to a MoinMoin Wiki server, then you will use the ePublisher Wiki - MoinMoin output format.

ePublisher supports the following types of Wiki servers:

- Confluence
- MoinMoin
- MediaWiki

If your organization has Wiki servers running, you can work with the IT support team in your company to identify a Wiki server on which you can deploy content. Based on your project needs and goals, your Wiki server may be one of the following types of Wiki servers:

- A Wiki server installed and managed by your corporate web team
- A Wiki server installed and managed by staff in a particular department in your company, such as a Wiki server managed by the documentation team or technical support team
- A Wiki server installed and managed by you

Defining the Wiki Content Maintenance Process

Carefully consider how you will maintain the content you deploy to a Wiki before you perform your first Wiki content deployment. As you define your Wiki content maintenance process, complete the following tasks:

- Identify how changes to Wiki content will be identified, reviewed, and managed. For more information, see “Reviewing and Managing Changes to Wiki Content”.
- Determine if changes to content on the Wiki will be incorporated back into source documents. For more information, see “Incorporating Changes Back Into Source Documents”.
- Identify how you will encourage content contributors to add content to your Wiki. For more information, see “Encouraging Content Contributors to Contribute”.

Reviewing and Managing Changes to Wiki Content

In most out-of-the-box Wiki installations, modifications to content on Wiki pages display almost instantaneously. There is no built-in content review workflow process to use when users make modifications to content published on a Wiki. If you plan to allow users to add their own information to Wiki pages, consider creating a process that will help you ensure that the added content is correct, appropriately categorized or labelled, and added in an appropriate logical structure on the Wiki. You should also have a process in place for relocating pages or deleting orphaned pages as needed. An **orphaned page** is a page with few or no links from other pages. For more information about orphaned pages and relocating or deleting orphaned pages, see your Wiki server documentation.

Most Wikis keep a record of changes made to Wiki pages. When a content contributor updates a Wiki page, the Wiki displays an entry in the recent changes list that shows the change made. For example, in Confluence, you can see recent edits or page additions in the Recently Updated area for a space. In MoinMoin, you can see recent changes on the Wiki on the Recent Changes tab. In MediaWiki, you can see recent changes on the Recent Changes page. This allows Wiki users to quickly and easily see when specific changes were made and by whom.

In addition to providing default pages to display recent changes, some Wikis can filter the recent changes list to remove minor edits and edits made by automatic importing scripts and content deployments. For more information about options for configuring the information displayed on the Recent Changes page for a Wiki, consult the documentation for your Wiki.

Many Wikis also allow users to include a short edit summary that describes the changes they made. The edit summary is not displayed on the Wiki page, but this information is retained as a part of the page history.

If you have a group of editors or reviewers that you want to review and approve changes to Wiki content, you will have to define your own processes for this workflow, and then configure the Wiki manually as needed to support this workflow. Some Wiki software allows users willing to maintain pages to be warned of modifications to the pages, allowing the user to verify the validity of new editions quickly. Some Wikis also allow you to set up an RSS feed so that recent changes can be monitored from an external application.

Most Wikis also provide simple revision control functionality that saves and stores a copy of every version of the page. Using the revision control functionality provided by the Wiki, users can see previous page versions and also use the **diff** feature to quickly and easily compare versions of a page and see precise changes made to a page. The **diff** feature highlights the changes between two versions and allows you to view and restore a previous version of the page as needed if a user mistakenly or maliciously adds incorrect or inappropriate content to a page.

Incorporating Changes Back Into Source Documents

An important point to consider before you deploy content to a Wiki is whether you plan to incorporate new or updated content from Wiki pages back into the source documents used by ePublisher to generate the Wiki pages.

ePublisher does not support the round-tripping of content, or generating Wiki pages using ePublisher, and then re-importing the Wiki pages back into ePublisher as source documents. Currently, changes made to content on Wiki pages must be manually re-entered into the source documents if you want to incorporate the changes made to Wiki pages into the next version of content ePublisher generates. This approach ensures content authors can thoroughly review and validate the content as needed for appropriateness and technical accuracy before inserting changes made to content on Wiki pages back into source documents.

If you do plan to try and incorporate some of the changes users make to Wiki pages back into source documents, typically a manager, content author, or other content owner identifies the additions or updates from Wiki pages that should be incorporated back into source documents using a change list. Writers then update the source documents as needed to include the new or updated content.

Encouraging Content Contributors to Contribute

Ensure you have a system in place for encouraging users to contribute comments, edits, and content. Nothing encourages contributions to Wiki content like feedback. Consider putting a system in place for acknowledging the contributions users add to your deployed Wiki pages as they are made. This could be a simple verbal thank you or a quick thank you on the Wiki page as soon as you see changes to content on the Wiki.

Defining Wiki Navigation

Users navigate between pages on a Wiki using hypertext links. In most Wiki implementations, users can create links as they add pages and content to the Wiki, and users are responsible for linking pages as they feel appropriate. As a result, by default most Wikis provide a generally flat navigational structure, rather than a more formal hierarchical structure, such as the hierarchical navigation typically provided by default in online help systems.

If you use ePublisher to generate and deploy Wiki content, you can use some of the navigational components provided by ePublisher to quickly and easily define a navigational structure for the Wiki content you deploy. For example, you can use ePublisher functionality to provide a basic hierarchical structure that users can use to help them navigate through ePublisher-produced content.

ePublisher can create and display the following navigational components on Wiki pages you generate and deploy with ePublisher:

- Table of contents (TOC) page
- Mini-TOCs on specified pages
- Navigation browse buttons at the top or bottom of Wiki pages
- Breadcrumbs at the top or bottom of Wiki pages
- Categories or labels for Wiki pages that help group pages with similar subjects

If you plan to use ePublisher to generate Wiki output, consider configuring ePublisher to create TOCs, mini-TOCs, navigation browse buttons, and breadcrumbs on your Wiki pages. TOCs and mini-TOCs provide useful navigational elements for users and are especially helpful for users viewing content on a Wiki. This is because Wikis, unlike more traditional online help systems, do not provide a default tri-pane hierarchical structure that can assist users when navigating content. Navigation browse buttons and breadcrumbs also help users to navigate through Wiki content.

When you use ePublisher to generate Wiki output, by default ePublisher places navigation buttons and breadcrumbs at the top of each page of your ePublisher-generated Wiki content. Stationery designers can quickly take advantage of these built-in navigational components by mapping table of contents levels in Style Designer and defining if and where ePublisher should create mini-TOCs in generated output and whether and where navigation buttons and breadcrumbs should display in Wiki output. For more information about defining TOCs and mini-TOCs, see “Defining TOCs and Mini-TOCs”. For more information about defining the appearance of your Wiki pages, including breadcrumbs and navigation browse buttons, see “Defining the Appearance of Pages”.

You can also use the WikiCategory marker to assign categories or labels to Wiki pages. Category or label tags on Wiki pages allow categorized pages to automatically be added to a list on a category page on the Wiki.

Note: MoinMoin and Media Wiki use the term *category* to describe page grouping functionality. Confluence uses the term *label* to describe page grouping functionality.

The category page lists all of the Wiki pages tagged for a certain category. For example, if you tag each page on a Wiki that contains licensing information with a Licensing category tag, then a licensing category page on the Wiki can display a list of all of the pages tagged as containing licensing information. For more information, see “Defining Wiki Categories or Labels”.

Understanding Wiki Search Capabilities

Studies have shown that among the current generation of Internet users, the first action users take when they need to find information is to use a search tool, like Google, rather than navigate through a series of steps or links.

Most Wikis support title-based searches, and many also include full-text search capabilities. The performance and scalability of searches can vary based on whether the Wiki server engine uses an underlying database. Several Wikis also support the integration of third-party specialist search engines such as Google.

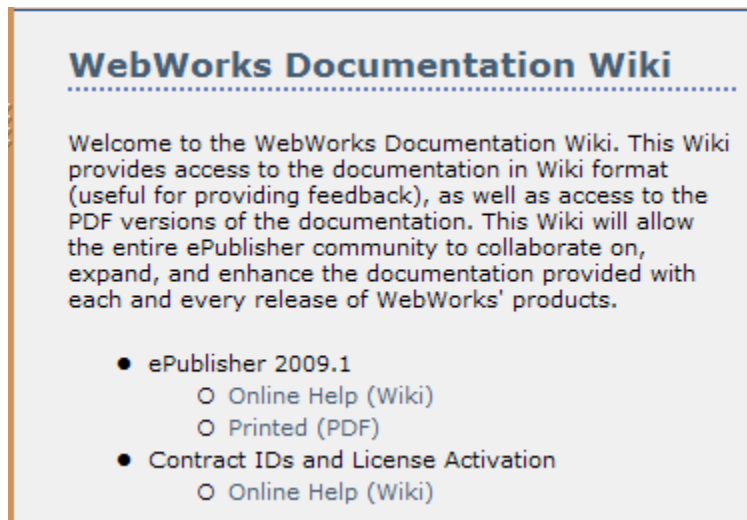
Some Wikis also allow you to customize the search features and functions available to users. For more information about the search capabilities your Wiki provides, consult your Wiki server documentation.

Creating Landing Pages

When you generate and deploy Wiki content, the highest entry point ePublisher provides is the table of contents that ePublisher automatically generates for the content when you define a TOC for your Wiki target. In order to make your Wiki content easily accessible to your users, consider creating one or more higher-level landing pages, or entry points, for the content you plan to deploy to a Wiki.

A **landing page** is simply a page you create on your Wiki that provides an overview of the content deployed to the Wiki. The landing page can include links to other special deployed Wiki pages that you want to highlight, such as the table of contents page generated by ePublisher, the first page of each major section of your deployed Wiki content, or any other content you want to highlight.

Following is an example of a landing page created on the WebWorks documentation Wiki at docs.webworks.com. This landing page introduces users to the WebWorks documentation Wiki and provides some entry-level links designed to help users easily access the deployed Wiki content.



You can create landing pages either before or after you generate and deploy Wiki output. However, designing your landing pages before you generate and deploy Wiki output is often useful, as it helps you think through all of the different entry points you may need to create for your Wiki output before you deploy it.

In MediaWiki and MoinMoin, you can create the page you want to use as your landing page simply by creating a new page on the Wiki and including the information and links to the Wiki content you plan to deploy as appropriate. With MediaWiki, you also have the option to use an existing namespace or create a new namespace to host your deployed Wiki output. For more information about creating pages in MediaWiki and MoinMoin, see the documentation for the Wiki server.

In Confluence, first create a space and a home page for your Wiki content, and then create the page you want to use as your landing page. For more information about creating spaces, home pages, and pages in Confluence, see the documentation for Confluence.

Defining and Configuring Wiki Permissions

A defining characteristic of Wiki technology is the ease with which pages can be created and updated. Generally, there is no review cycle when users edit wiki pages or add or delete pages. You can create a Wiki that is open to the general public that allows users to make changes without registering, or you can limit access to Wiki content to specific user groups and allow only those users view and edit permissions. You can also configure permissions on your Wiki to allow users to view but not edit or change Wiki content. Private Wiki servers typically require user authentication to edit pages, and sometimes even to read them.

Depending on the type of Wiki, the Wiki administrator can configure different access and editing privileges for users and groups. For example, MoinMoin supports hierarchical access rights, while MediaWiki and Confluence allow you to configure access based on namespace (for MediaWiki) or space (for Confluence) or page (for both MediaWiki and Confluence).

By default, most Wikis do not require users to register before viewing content. As you plan your content deployment to a Wiki, determine if you want to allow all users to view your content, or if you want to configure access so that only certain users or groups of users are able to view the content you deploy to a Wiki. For example, in a corporate environment, you may want to allow all internal employees to view specific pages on your Wiki, but limit the ability of contractors to view certain pages. You may also want to allow a group of internal employees to edit some Wiki pages but allow only customers to view Wiki pages.

Also ensure you identify the users or groups of users you want to allow to view, update, and create Wiki pages. Most Wikis require that anyone who wants to add or update Wiki content first register as a user on the Wiki. Once users register, the Wiki can create a signature cookie at the start of an editing session and track the edits the user makes.

For example, on a MoinMoin Wiki, Wiki pages display as *Immutable* if the user is not logged in, and the user cannot make any changes to the page. Once the user logs in, MoinMoin tracks the changes the user makes and displays the changes in the Recent Changes area of the Wiki.

Once you have defined appropriate user permissions for content on your Wiki, configure the permissions on the Wiki. For more information about configuring Wiki permissions, see the documentation for your Wiki server.

Verifying Your Wiki Server Configuration

Before you generate and deploy Wiki content, verify that your Wiki server is configured correctly. If your Wiki server is not configured correctly, ePublisher may not be able to deploy content to the Wiki. For more information about Wiki server configuration requirements, see the following topics:

- “Wiki - Confluence”
- “Wiki - MediaWiki”
- “Wiki - MoinMoin”

Generating and Deploying Wiki Content

Wiki content generation and deployment follows the same general process used to generate and deploy content using other format types that ePublisher supports, such as online help and PDF.

Use the following steps to help you understand this process:

1. Identify where you will deploy your Wiki content. For more information, see “Identifying Your Wiki Server and Wiki Output Format”.
2. Ensure that the Wiki where you will deploy your Wiki content is configured correctly. For more information about specific configuration requirements for the type of Wiki to which you plan to deploy content, see “Wiki - Confluence”, “Wiki - MediaWiki”, and “Wiki - MoinMoin”.
3. Identify an existing login on the Wiki that ePublisher can use when deploying content or create a new login on the Wiki for ePublisher to use to deploy content. For more information about creating a login on the Wiki, see the documentation for your Wiki server.
4. *If you have an existing Stationery design project*, add a target that uses the Wiki output format that you want to use for your existing Stationery design project. For more information, see “Adding Output Formats to Your Stationery Design Project” and “Adding a Target to Your Stationery Design Project”.
5. *If you do not have an existing Stationery design project*, create a new Stationery design project and add a target that uses the Wiki output format for your new Stationery design project. For more information, see “Creating a Stationery Design Project”, “Adding Output Formats to Your Stationery Design Project” and “Adding a Target to Your Stationery Design Project”.
6. Specify the appearance and other features you want your deployed Wiki content to have in the Wiki target in the Stationery design project. Ensure you define the following items for your Wiki target
 - Page breaks. For more information, see “Defining New Pages (Page Breaks)”.
 - TOCs and Mini-TOCs, including the appearance of mini-TOC elements. For more information, see “Defining TOCs and Mini-TOCs”.
 - Appearance of paragraphs, lists, characters, images, pages, and links for Wiki output. For more information, see the following topics:
 - “Modifying the Appearance of Paragraphs” and “Adjusting Paragraph Styles for Wiki Markup”
 - “Defining the Appearance of Bulleted and Numbered Lists for Wiki Markup”
 - “Modifying the Appearance of Characters”
 - “Defining the Appearance of Tables”
 - “Defining the Appearance of Images”
 - “Defining the Appearance of Pages”.
 - “Defining the Appearance of Links”

You can also choose to define output file names and specify categories for your Wiki pages using Adobe FrameMaker markers, Microsoft Word field codes, and `@otherprops` attributes from DITA, which come through as markers. For more information, see the following topics:

- “Defining File Names”
- “Defining Wiki Categories or Labels”

7. Specify target settings as appropriate.

For example, you can specify how you want to deploy **baggage files**, or files placed in the `Files` folder for the target. If you are generating and deploying MediaWiki output, you can specify if you want to use a prefix for each generated MediaWiki page.

For more information about specifying target settings for Wiki output, see “Specifying Output Format-Specific Settings”.

8. Create an output destination in ePublisher that specifies where you will deploy your generated Wiki output files.

Note: You must create a deployment point and deploy generated Wiki content before you can view the Wiki content.

For more information, see the following topics:

- “Understanding Output Deployment” and “Creating Output Destinations”
- “Specifying Output Destinations for Targets”
- “Deploying Output to Output Destinations”

9. Generate output for your Wiki target. For more information, see “Generating Output”.

10. Deploy the Wiki output files to a Wiki server. For more information, see “Deploying Output to Output Destinations”.

Note: You must deploy generated Wiki content to a Wiki server before you can view the Wiki content.

11. Verify that your Wiki output has the appearance you want. Adjust settings in the Stationery design project as needed and then regenerate and deploy Wiki output files as needed until you obtain the look you want for your Wiki output.

12. Save your Stationery based on your Stationery design project and test the Stationery. Verify that the Wiki output files generated with the Stationery display appropriately on the Wiki server and in the appropriate location. For more information, see “Saving and Testing Stationery”.

13. Verify that users can access your deployed Wiki content from any landing pages you created on the Wiki that link to your generated Wiki output. For more information about landing pages, see “Creating Landing Pages”.

14. Distribute the Stationery that supports the appropriate Wiki output formats to content developers responsible for generating Wiki output. Also include the following information:

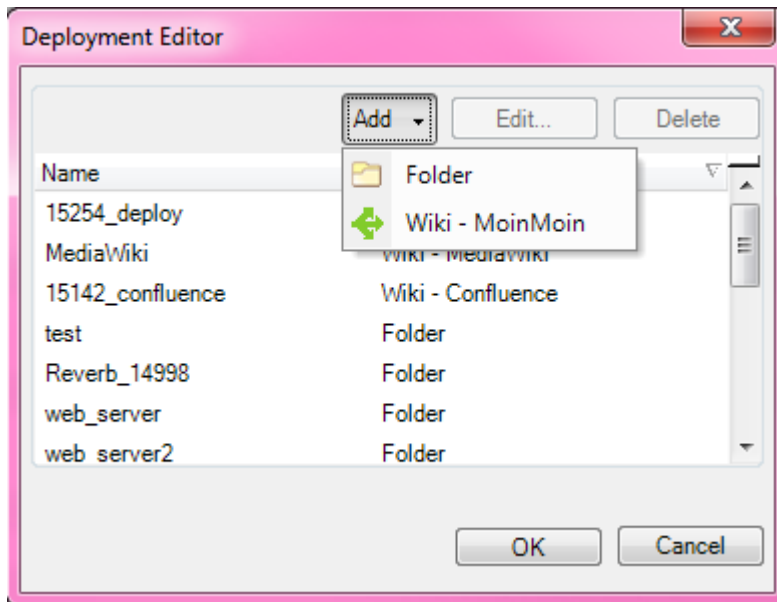
- Information about the Wiki servers, logins for ePublisher to use when deploying Wiki content, and locations on Wiki servers where they can deploy content.
- Information about any existing entry points on the Wiki for their content, as well as information about how they can create additional entry points for their Wiki content, if needed.
- Information about how the content deployed on a Wiki will be viewed, updated, and maintained after deployment.

Wiki Deployment - Start to Finish

The following section provides a basic walkthrough of deployment to a Wiki server.

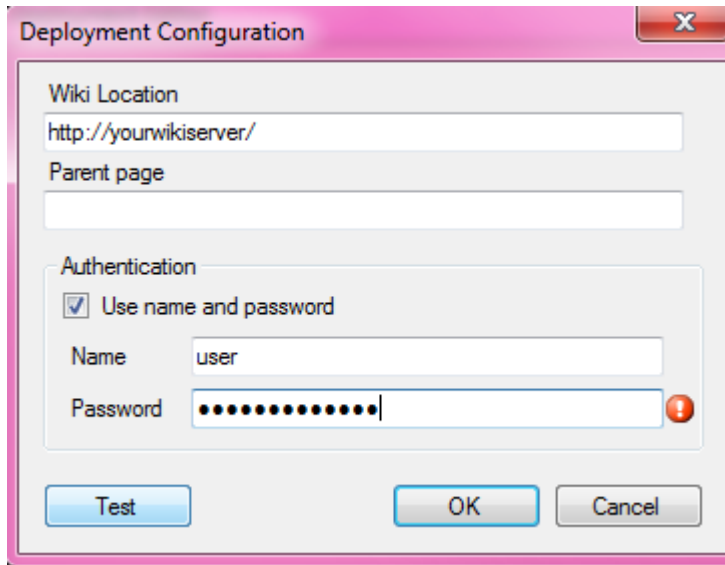
To deploy ePublisher Output to a Wiki server please complete the following steps:

1. Create a new project or open an existing ePublisher Designer project
2. Go to Target -> Manage Targets
3. Click Add
4. Name the Target -> Select the Wiki Format from the dropdown, hit OK, and close the Manage Targets Window
5. After you have configured the customizations appropriate to the project, make sure the Wiki Target is selected in the dropdown from the Toolbar
6. Generate Output using Generate Selected or Generate All from the tool bar, or the context menus
7. Go to Target -> Target Settings and click Add Deploy Target button
8. Click the Add button to bring up the Wiki selection



Note: The specific wiki deployment option will change depending on the currently active target. For example, you may only define “Wiki - MoinMoin” deployment targets when a “Wiki - MoinMoin” target is active.

9. In the Deployment Configuration, select a unique name, and then click Edit
10. In the following window, you will add the information as supplied to you by the System Administrator of the Wiki:



11. You can test the connection to the Wiki Server by clicking Test, if successful there will be an alert Window that says “Success!”, if unsuccessful there will be a red exclamation as shown in the image above
12. After you have tested, you will click Ok, and this should bring you back to the Deployment editor, where you will see your newly created Deployment Configuration
13. Click Okay to see the Deploy Target in the Target Settings, and click Okay to get back to the ePublisher main window
14. Click Target -> Deploy to upload the generated output to the server
15. Navigate to the wiki server in the browser and go to the Recent Changes area to verify that the files have been uploaded

Wiki WYSIWYG Support

WYSIWYG editor support is determined by the wiki server, not ePublisher. Additionally, each wiki has different capabilities regarding WYSIWYG support.

- MediaWiki - offers no WYSIWYG support. When the user clicks edit, the markup appears but not as it would be on the screen
- MoinMoin - offers WYSIWYG support for Firefox and Internet Explorer
- Confluence - Depends on the version in use. Versions prior to 4.x offer WYSIWYG for most modern browsers. Starting with version 4.0, only WYSIWYG editing is offered. WYSIWYG editing may be limited due to wiki markup limitations in Confluence 3.x and earlier. Confluence 4.x WYSIWYG editing is limited by the fact that ePublisher emits Confluence wiki markup rather than native HTML. Please consult the Confluence documentation for any further questions regarding compatibility at <http://confluence.atlassian.com>

Integrating ePublisher Online Help with the WebWorks Documentation Wiki

The ePublisher product team has integrated the ePublisher online help with the WebWorks documentation Wiki. This integration provides the following benefits:

- WebWorks employees, partners, and customers can add comments, edits, and content to the ePublisher topics on the docs.webworks.com Wiki in real time, improving, enhancing, and extending the ePublisher documentation as needed instantaneously.
- ePublisher users with an Internet connection who view the ePublisher online help system installed on their computers can quickly see the following information:
 - If a page they are viewing in the ePublisher help system on their local computer has been updated on the Wiki
 - If other users have added comments about the page on the Wiki

For more information about how this integration appears to ePublisher users, see “Contacting Quadralay”.

This section contains information about how the ePublisher product team performed the integration. If you want to perform a similar integration, understand that your steps may be different based on the following combination of factors:

- How you generate your content
- The output formats you use
- The type of Wiki server you use
- How you configured your Wiki server

The information provided here is provided only as example information. It is intended to help advanced ePublisher Stationery designers who are comfortable performing complex ePublisher Stationery designs and customizations better understand the approach the ePublisher team used when integrating the ePublisher online help system with the docs.webworks.com Wiki.

To integrate the ePublisher online help system with content on the docs.webworks.com Wiki, the ePublisher team performed the following actions:

- Created an ePublisher WebWorks Help target that used the WebWorks Help output format.
- Created a docs.webworks.com Wiki target that used the Wiki - MoinMoin output format.
- Specified that ePublisher use the same page naming pattern for each target. This ensured that a content page generated for the WebWorks Help output used the same name as the corresponding page generated for the Wiki output. For more information about specifying file names for generated output, see “Defining File Names”.
- Created a `status.html` page and supporting image (`.gif`) and text (`.txt`) files. These files render the **Wiki** and **Discussion** links and radio buttons displayed in the top right corner of each page of the ePublisher online help system using a simple table and graphical layout. For more information about the `status.html` page and to see an example of a `status.html` page, see wiki.webworks.com/DevCenter/Projects/WikiFeedback.

- Created an AJAX JavaScript file to place on the docs.webworks.com Wiki server. The computer running the ePublisher online help system locally uses this script to obtain information about page status on the docs.webworks.com Wiki server. For more information about the AJAX JavaScript file and to see an example of the AJAX JavaScript, see wiki.webworks.com/DevCenter/Projects/WikiFeedback.
- Embedded an iframe element in the upper right corner of each page in the WebWorks Help system by modifying the `Page.asp` file. The iframe element displays the `status.html` page and its associated images and displays the **Wiki** and **Discussion** links and radio buttons. The iframe element also communicates with a page on the docs.webworks.com Wiki server that queries for updates on the Wiki server.

The iframe is an inline frame from a frame set that allows you to embed a document within another document. The ePublisher product team embedded an iframe on each page of the ePublisher online help set in order to give the pages in the ePublisher online help system a way to communicate with the docs.webworks.com Wiki. Without the iframe, the pages in the ePublisher online help system cannot communicate with the docs.webworks.com Wiki because they are simply HTML files running on the local file system. The iframe element is required in order to provide the security context needed for communication back to the docs.webworks.com Wiki.

Following is an example of the iframe element added to the `Page.asp` file for the ePublisher online help system:

```
<iframe src="http://docs.webworks.com/webworks_static/status/2009.1/status.html?/
FrontPage" wwpag:attribute-src="wiki-status-url" width="96" height="43" frameborder="0"
scrolling="no">
</iframe>
```

For more information about working with `Page.asp` files, see “Customizing the Content Page Design”.

When users access a page in the ePublisher online help system, the following actions occur:

1. The iframe element on each page of the ePublisher online help system communicates back to `status.html` page on the docs.webworks.com Wiki.
2. The `status.html` page calls the AJAX JavaScript on the docs.webworks.com Wiki to do an AJAX request to determine if there is a difference between the contents of the page in the online help system and on the docs.webworks.com Wiki.
3. If there is a difference, then the images presented in the frame on the ePublisher online help system page are updated to indicate new information is available.
4. Users can click the appropriate link in the ePublisher online help system to see the updated Wiki content or add a comment to the page on the Wiki.

The ePublisher online help and docs.webworks.com Wiki integration provides simple yet powerful example of how you can link and integrate an online help system with a Wiki, leveraging the easy-to-use, approachable feedback mechanisms built into Wiki software to update, enhance, and extend your content in real time.

8

Designing PDF Output

With ePublisher, you can quickly and easily create PDF output from Adobe FrameMaker, Microsoft Word, and DITA-XML source documents. Based on your requirements, you can either create PDF output so that it mirrors the styling and layout of your source document's pre-configured settings (using the *PDF* output format), or you can choose to manage all these settings through ePublisher using the *PDF - XSL-FO* output format.

Why use PDF - XSL-FO output format?

If you want to keep your original document's style and layout settings, then you will want to configure ePublisher to use the *PDF* output format. However, if you want to directly control your document's style and layout settings, then you will need to configure ePublisher to use the *PDF - XSL-FO* output format.

PDF XSL-FO Font Inclusions

To ensure non-standard fonts are included into your generated PDF, you may need to modify the Apache FOP configuration files, `apache-fop.xconf` or `apache-fop-1.0.xconf`.

To do this, you must either create a Format or Target Override, Create the override in one of these places, for more information on overrides, See “Project Format and Target Overrides”. Depending on how you prefer to set up your project, you will create one of the following:

Per target:

```
Targets\<target name>\Helpers\apache-fop.xconf Targets\<target name>\Helpers\apache-fop-1.0.xconf
```

Per format:

```
Formats\<format name>\Helpers\apache-fop.xconf Formats\<format name>\Helpers\apache-fop-1.0.xconf
```

Project-wide:

```
Formats\Helpers\apache-fop.xconf Formats\Helpers\apache-fop-1.0.xconf
```

Once the override has been created, open the file in a text editor of your choosing, and you will see the following markup:

```
<fonts>

<!-- Example -->

<!--

<font kerning="yes" embed-url="file:///C:/Windows/Fonts/REFSAN.TTF">

<font-triplet name="Microsoft Sans Serif" style="normal" weight="normal" />

</font>

<font kerning="yes" embed-url="file:///C:/Windows/Fonts/REFSAN.TTF">

<font-triplet name="Microsoft Sans Serif" style="italic" weight="normal" />

</font>

-->

<!-- automatically detect operating system installed fonts -->

<auto-detect/>

</fonts>
```

Modify the lines in between the comments to reflect the font you are wanting and the directory path in which it is located. Save this file, and now you can add them by using ePublisher Designer.

To add fonts in ePublisher Designer

1. In ePublisher Designer, select the **View -> Style Designer** menu
2. With the **Paragraph Styles** visible, select the **[Prototype]** style
3. On the **Properties** tab, select **Font** then click the icon for **Family**
4. If your fonts are not listed, then manually add the names of your desired font using the **Custom Font Family** text box
5. Select **OK**

Custom Header and Footers

One of the advantages of selecting this format over the traditional PDF format is the ability to customize the footer and header depending on the Page Styles in the Style Designer. Please keep in mind that this will only be available per ePublisher Designer as the Style Designer is not available in ePublisher Express.

To customize the Header or Footer

- 1. In ePublisher Designer, go to the **Style Designer -> Page Styles**
- 2. Select the Page Style that you have defined and click the **Options** tab
- 3. Depending on where you would like to have your information you can enter in information for Even or Odd pages
- 4. Chose to Enable or Disable Header and Footer text under the Generate Output option
- 5. Chose from the following Variables to customize the content

\$Title;	This variable expands to the title of the PDF. For the document-level PDF, this value is the title of the document or the value of the first paragraph. For the group-level PDF, this value is the name of the Group. For the project-level PDF, this value is the name of the Target
\$PageNumber;	This variable expands to the current page number
\$RunningTitle;	This variable expands to the text of the last paragraph in the flow of the document which generated a TOC entry
\$ChapterTitle;	This variable expands to the text of the top-most TOC entry of the document.

9

Designing ePUB Output

ePublisher can deliver content to your mobile users and eReader platforms with the IDPF ePUB 2.0 eBook standard. This format makes sense for long form content or reference materials which are not suitable for web delivery. eBooks enable users to access content offline, track their reading progress, and otherwise enjoy the benefits of traditional press books.

ePUB Platforms

ePUB eBooks can be accessed and viewed on a variety of desktop and mobile devices. To ensure maximum usability, this release was tested against the following ePUB readers:

- Adobe Digital Editions
<http://www.adobe.com/products/digitaleditions/>
Microsoft Windows
- Apple iBooks
<http://itunes.apple.com/us/app/ibooks/id364709193?mt=8>
Apple iPad, Apple iPhone, Apple iPod Touch
- Lexcycle Stanza (Apple iPad, iPhone, and iPod Touch)
<http://www.lexcycle.com/iphone>
Apple iPad, Apple iPhone, Apple iPod Touch, Microsoft Windows (Beta)

In addition UX tests performed on these eBook platforms, generated output was validated using the publicly available EpubCheck tool during the development cycle.

- EpubCheck 1.0.5
<http://code.google.com/p/epubcheck/>

By closely tracking the ePUB standard and conducting eReader specific tests, ePublisher attempts to deliver high quality eBooks across a wide range of ePUB capable platforms.

ePUB Considerations

The ePUB standard specifies a container and page definition syntax. ePUB eBook readers will render this information as best they can given available screen real-estate and platform considerations. Users who wish to maximize the ePUB reading experiences for a broad audience should pay careful attention to ePUB specific details.

Meta Data

You can specific standard ePUB eBook meta data in the Target Settings dialog.

- Author Name
- Author Name (File As)

Book Title and ID

All ePUB books contain a unique identifier. ePublisher will synthesize a unique ID for your book, though you may require explicit control over this value. You can specify an explicit ID via the Merge Settings dialog with the Group Context control. Further, localized book titles can be specified in the Merge Settings dialog as well.

Long Content

The ePub standard recommends breaking long content into smaller chunks to speed display rendering and reduce memory requirements on mobile devices. This can be accomplished with ePublisher's standard page break controls. You may specify a page break priority for paragraphs and tables in the Options panel of the ePublisher Style Designer.

Page Styles

Content page styles can be controlled using ePublisher's standard page style support. For the ePUB format, ePublisher also allows users to select a page style for use with the cover and index pages. These styles can be specified in the Target Settings dialog under the Cover and Index sections.

Tables

By default, ePublisher's ePUB format renders tables using paragraph markup. Users may enable true table markup via the table styles Options panel in the ePublisher Style Designer. Reasons for rendering tables using paragraph markup include:

- All tested eBook readers lacked the capability to view tables larger than the available screen space.
- The Apple iBooks reader, prior to version 2.0, fails to render images inside of table cells.
ePublisher emits conversion warnings when an image is rendered inside a table cell.

Tables can be used effectively inside of ePUB eBooks, though authors should be aware of the inherent space limitations associated with mobile reading devices. The addition of proportional (percentage based) table cell widths, used in conjunction with a table width set to 100%, may provide users with sufficient control over table rendering behavior. The behavior can be specified in the Options tab for table styles in the ePublisher Style Designer.

Cover

ePUB books support the display of a cover page or image. This format allows users to select a cover image via the Target Settings dialog. Additionally, users may customize the cover page by creating overrides for the `Pages\Cover.asp` page template.

When creating your cover page and image, keep the following points in mind:

- eBook icons
 - Adobe Digital Editions uses the cover page for the eBook icon
 - Apple iBooks and Lexcycle Stanza use the cover image for the eBook icon
 - ePubublisher will embed the selected cover image into the cover page to ensure correct operation across platforms
- Cover image limitations
 - Certain readers, such as the Apple iBook reader, prefer images with a 3x4 aspect ratio. Otherwise, the eBook will not display a custom icon in iTunes.
 - Certain readers, such as the Apple iBook reader, prefer 600x800 pixel images. Otherwise, the eBook will not display a custom icon in iTunes.

Syncing with Apple iPad

1. Ensure iTunes 9.1 or greater is installed on your system. Ensure the Apple iBooks application is installed on your iPad.
2. Launch iTunes.
3. Click “Add File to Library” and select your generated ePUB eBook.
4. Attach your iPad.
5. Sync your iPad normally.
6. When synchronization completes, disconnect the iPad from your computer.
7. Launch the iBooks application on the iPad.
8. Within iBooks, you will see your new book on the virtual bookshelf.

These instructions also apply when adding ePUB eBooks to Apple iPhone and Apple iPod Touch devices.

7

Producing Output Based on Stationery

This section explains how writers can use their source documents and ePublisher projects and Stationery to produce output.

Checklist: Producing Output Based on Stationery

Use the following checklist to help you use your source documents and ePublisher projects and Stationery and to produce output. For more information about designing Stationery, see “Designing Stationery”.

<input checked="" type="checkbox"/>	Task
<input type="checkbox"/>	1. Review the conceptual information related to projects, source documents, targets, and Stationery. For more information, see “Understanding Projects and the Project Folder Structure”, “Understanding Source Documents”, “Understanding Targets”, and “Understanding Stationery”.
<input type="checkbox"/>	2. Talk to the Stationery designer to determine what Stationery you should use to generate output.
<input type="checkbox"/>	3. Identify the source documents you want to use to generate output.
<input type="checkbox"/>	4. Prepare your source documents. For more information, see “Preparing Adobe FrameMaker Source Documents” and “Preparing Microsoft Word Source Documents”.
<input type="checkbox"/>	5. Create a project using the Stationery created by the Stationery designer. For more information, see “Creating Projects Based on Stationery”.
<input type="checkbox"/>	6. Add source documents to your project. For more information, see “Adding Source Documents to Projects”.
<input type="checkbox"/>	7. If you are generating WebWorks Help output , replace the default WebWorks Help splash image. For more information, see “Customizing or Removing Splash Page Images in WebWorks Help”.
<input type="checkbox"/>	8. Generate output. For more information, see “Generating and Regenerating Output”.
<input type="checkbox"/>	9. View your output. For more information, see “Viewing Output”.
<input type="checkbox"/>	10. If you want to use reports to validate your output , review reports. For more information, see “Validating Output Using Reports”.
<input type="checkbox"/>	11. If you want to merge multiple help systems into one help system , configure merge settings for your merged help system. For more information, see “Merging Help Systems (Multivolume Help)”.
<input type="checkbox"/>	12. If you want to customize target settings , customize target settings as needed. For more information, see “Customizing Target Settings”.
<input type="checkbox"/>	13. If you want to customize variable and condition settings , customize variable and condition settings as needed. For more information, see “Customizing Variable Settings in Projects” and “Customizing Condition Settings in Projects”.
<input type="checkbox"/>	14. If you want to customize cross reference settings , customize cross reference settings as needed. For more information, see “Customizing Cross-Reference Settings in Projects”.
<input type="checkbox"/>	15. Deploy your output. For more information, see “Deploying Output”.

Understanding Projects and the Project Folder Structure

This section explains what a project is and the folder structure projects use.

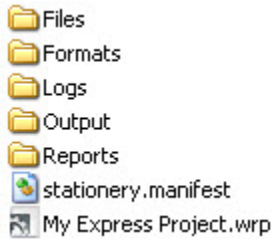
Understanding Projects

An ePublisher project consists of all the necessary pieces needed to convert your source documents into online output. It contains your source documents, images, project Stationery, and any settings or preferences you specify. After you create a project, you can modify your project settings and preferences, generate reports, and produce output.

ePublisher provides powerful single-sourcing capabilities that allow you to generate online output in multiple formats using a single project and a single set of source documents. For example, you can generate WebWorks Help, Microsoft HTML Help, Oracle Help, Dynamic HTML Help, and Sun JavaHelp using one project and one set of source files.

Understanding the Project Folder Structure

The following figure shows a sample project folder.



To view all of the files for your project, on the **View** menu, click **Project Directory**.

The project folder contains the following subfolders:

Files

Contains any custom files you want your project to use. Typically, the `Files` folder contains logo images, custom `.css` files, custom bullet images, and custom background images. To view the files in the `Files` folder for your project, on the **View** menu, click **User Files**.

You can also place a `Files` directory in a target's override folder or within the target's format override folder.

- `Formats\<format name>\Files`
- `Targets\<target name>\Files`

These files will be copied for each target's sub-directory but will not be visible in the user interface to select, e.g. the company logo image in the Target Settings dialog.

Formats

Contains output format overrides and all of the files required to generate output for an output format. Any time you want to override an output format file, place the override in the `Formats` folder. For example, if you want to override the standard table of contents icons for topics in WebWorks Help by specifying custom table of contents icons, place the custom table of contents icons you want to use in an `images` folder in the `Formats` folder. For more information about format overrides, see "Creating Format Overrides". To view the files in the `Formats` folder for your project, on the **View** menu, click **Format Override Directory**.

Logs

Contains the `generate.log` file. The `generate.log` file contains information about the actions ePublisher performed when generating output, along with any warning or errors that occurred during output generation.

Output

Contains the files ePublisher creates when generating output and provides a structure for generated output files. ePublisher creates a folder for each target in the project in the `Output` folder, and each target folder contains all topic pages, generated images, entry-point files, and merged help files generated for the target. The **entry-point file** is the file that opens the help system.

By default, ePublisher creates the `Output` folder in the following location:

- *If you use ePublisher Express*, by default ePublisher creates the `Output` folder in the `My Documents \ePublisher Express Projects\ProjectName` folder, where *ProjectName* is the name of the project.

- **If you use ePublisher Designer**, by default ePublisher creates the `Output` folder in the `My Documents\ePublisher Designer Projects\ProjectName` folder, where `ProjectName` is the name of the project.

You can view output files in the `Output` folder using Output Explorer or by clicking on and opening output files in the `Output` folder. To view the files in the `Output` folder for your project, on the **View** menu, click **Output Directory**. For more information about viewing output, see “Viewing Output in Output Explorer” and “Viewing Output in the Output Folder”.

Reports

Contains all of your configured reports in an XML format that ePublisher can display.

Targets

Contains target overrides. Any time you want to override a file in a target, place the override in the `Targets` folder. ePublisher looks for overrides in the `Targets` folder before it looks for overrides in the `Formats` folder. The `Targets` folder only appears in your project folder if your Stationery is configured to use target overrides. For more information about target overrides, see “Creating Target Overrides”. To view the files in the `Targets` folder for the selected target in your project, on the **View** menu, click **Target Override Directory**.

Your project also uses a `Data` folder. The `Data` folder contains information about how files in your project have been processed. The `.wif` files, which are located in the `Data` folder, contain style and content information from your source documents. ePublisher creates the `Data` folder in the following temporary folder location:

`Documents and Settings\UserName\Local Settings\Temp\WebWorks\ePublisherComponent\Data`

where `UserName` is the name of the ePublisher user, and `ePublisherComponent` is the name of the ePublisher component used to generate output, such as ePublisher Express or ePublisher Designer.

To view the files in the `Data` folder for your project, on the **View** menu, click **Data Directory**.

Understanding Source Documents

Source documents are documents you create your content in using a content authoring tool such as Microsoft Word, Adobe FrameMaker, or an authoring environment that supports DITA authoring, such as Adobe FrameMaker, XMetaL, or other XML content authoring tool. After you prepare your source documents, you use your source documents to generate output. For more information about preparing your source documents, see the *ePublisher Writer Guide*. For more information about generating output, see “Generating and Regenerating Output”.

Understanding Baggage Files

Baggage files are PDF, HTML and ZIP files that are not part of the ePublisher source files to be converted. ePublisher makes this determination by examining the file extension. If the file extension matches one of the following listed below then it's considered a baggage file.

```
.pdf  
.html  
.htm  
.shtml  
.shtm  
.xhtml  
.xhtm  
.zip
```

In order to a file to be processed as a Baggage file ePublisher must know about it. This can be achieved in one of two ways.

- (1) Any link from a source file being converted by ePublisher.
- (2) Entry in the `baggage file info list`. For more information see “Baggage files info list”.

Baggage Files will be packaged within your Output folder.

Note: In **Target Settings > Links > Baggage File Target**, specify whether you want the link to open in the same browser window or in a different window. There are several other options, see “Baggage File Target”.

Note: Set **Target Settings > File Processing > Insert Mark of the Web (MOTW) = Disabled**. Otherwise, the link will fail in Internet Explorer on the local file system. It will work in other browsers. It will work in IE on a web server.

Note: Baggage files will only be generated if you have the **Client-side Search** option ON in **WebWorks Reverb**, see “Client-side Search”.

Understanding Targets

A **target** is the specific type of output you want to produce using your source files and project settings. Targets are based on the output formats you specify for your project, and include all of the project settings you specify for each output format included in your project when you configure your project.

For example, assume that you are a writer working at CompanyA, and you have the requirement to create a web-based help system using your source documents. In this scenario, you create a project using stationery that supports WebWorks Help output and then create a target called CompanyA WebWorks Help.

Next, assume that your documentation requirements change, and in addition to creating WebWorks Help for CompanyA, you must now also produce Microsoft HTML Help and PDF files for CompanyA using your same source documents. In this scenario, you update your project to now include the following targets:

- CompanyA WebWorks Help
- CompanyA Microsoft HTML Help
- CompanyA PDF Files

Finally, assume your documentation requirements change again, and now, based on an Original Equipment Manufacturer (OEM) agreement your company signed, in addition to creating WebWorks Help, Microsoft HTML Help, and PDF files for CompanyA, you must use your same set of source documents to create WebWorks Help, Microsoft HTML Help, and PDF files for CompanyB, using company information and variables and conditions specific to CompanyB. In this scenario, you update your project to now include the following targets:

- CompanyA WebWorks Help
- CompanyA Microsoft HTML Help
- CompanyA PDF Files
- CompanyB WebWorks Help
- CompanyB Microsoft HTML Help
- CompanyB PDF Files

When you have multiple targets included in a project, you choose an active target and then specify project settings for the active target. The active target is the target currently selected in your project. When you want to modify project settings for an target, if you have multiple targets included in your project, ensure you have the correct target selected in the project when you modify project settings. For more information about specifying an active target, see “Specifying Active Targets”.

Understanding Stationery

The Stationery designer creates Stationery with ePublisher Designer using a Stationery design project. Stationery specifies the settings ePublisher uses to generate output. Stationery designers create Stationery by creating a Stationery design project in ePublisher Designer and then saving the processing rules, styles, and other information specified in the Stationery design project as Stationery. ePublisher Express and ePublisher AutoMap can then use the Stationery to generate output.

When the Stationery designer creates a project in ePublisher Designer and then saves a project using the **Save As Stationery** option, ePublisher Designer creates a Stationery file. A Stationery file is a file with the `.wxsp` file extension that contains formatting, project settings, project overrides, and style information. Source documents and document-specific information, such as Document Manager groups, are not saved in Stationery. After the Stationery designer creates the Stationery, writers use the Stationery provided by the Stationery designer when they create an ePublisher Express project. Writers use their ePublisher Express project and Stationery to generate output.

When the Stationery designer saves the Stationery, ePublisher creates the following folders:

- `StationeryName\Formats\OutputFormat`
- `StationeryName\Formats\OutputFormat.base`

where `StationeryName` is the name the Stationery designer specified for the Stationery, and `OutputFormat` is the type of output format the Stationery Designer specified for a target in the Stationery.

The `StationeryName\Formats\OutputFormat` folder contains any customizations or overrides the Stationery designer specified when designing the Stationery. ePublisher Express synchronizes with the files in the `OutputFormat` folder and uses the information about customizations and overrides contained in files in the `OutputFormat` folder to generate output.

Note: The Stationery may have one or more `OutputFormat` folders, based on the settings the Stationery designer specified.

The `StationeryName\Formats\OutputFormat.base` folder contains copies of all the files located in the `\Program Files\WebWorks\ePublisher\2018.1\Formats\OutputFormat` folder. These files define the default output format and transforms and are installed by default when you install ePublisher.

Stationery designers can do a compare, or **diff**, between the files located in these folders to quickly see any customizations or overrides specified for the Stationery. Stationery designers can use this information to help them reapply customizations and overrides as needed when designing a newer version of the Stationery in ePublisher Designer.

When the styles or features used in the generated output need to change, the Stationery designer uses the ePublisher Designer Stationery design project to update the styles and features specified in the Stationery, and then the Stationery designer saves the changes, creates updated Stationery, and deploys the Stationery. Once the new Stationery is available, writers synchronize their ePublisher Express project with the updated Stationery file and use the updated Stationery the next time they generate output.

Note: When you synchronize your project with Stationery, the synchronization process overwrites any target setting customizations you configured for the project.

For more information about synchronizing Stationery, see “Synchronizing Projects with Stationery”. For more information about designing and deploying Stationery, see “Designing Stationery” and “Deploying Stationery”.

Creating Projects Based on Stationery

Writers use ePublisher Express and Stationery created by a Stationery designer to generate output. When you use ePublisher Express to create a project based on Stationery, you specify the Stationery you want the project to use and the source documents you want to include in the project. The Stationery file uses a `.wxsp` file extension and contains information and settings for the project to use, such as style or format information, variable values, condition settings, cross-reference definitions, and more. The source documents contain the content for which you want to generate output. The project uses the settings specified in the Stationery file and the content and formatting in the source documents to generate output. A project file created with ePublisher Express uses the `.wrp` file extension.

Note: You cannot create a project based on Stationery using ePublisher Designer. You can only create projects based on Stationery using ePublisher Express. Stationery designers use ePublisher Designer to create Stationery using Stationery design projects. For more information about creating Stationery and Stationery design projects, see “Creating a Stationery Design Project” and “Saving and Testing Stationery”.

To create a project based on Stationery

1. In ePublisher Express, on the **File** menu, click **New Project**.
2. In the **Project Name** field, type a name for your project.
3. In the **Location** field, specify the location where you want to save your ePublisher project by clicking on the folder icon and browsing to the location where you want to save your project.

Note: Ensure you consider the length of the full path you specify for the project name and location. If you specify long names and paths for project, Windows may not be able to support the length of the full path.

By default, ePublisher stores projects in the `My Documents\ ePublisher Express Projects` folder.

4. In the **Standalone stationery** field, specify the Stationery you want to use to create your project by clicking on the folder icon and browsing to the location of the Stationery file.
5. Select a Stationery file (`.wxsp` file), and then click **Open**.
6. Click **Next**.
7. Click **Add**.
8. Browse to the location of the source documents you want to include in your project, select the source documents, and then click **Open**.

Note: You can add source documents when you create your project or you can add source documents after you create your project. For more information about adding source documents to projects, see “Adding Source Documents to Projects”.

9. Click **Finish** to create the project. ePublisher creates the project and gathers information about the structure of your source documents.

After you create your project, add targets to your projects as needed and then generate output. For more information, see “Adding Targets to Projects Based on Stationery” and “Generating Output”.

Working with Source Documents

This section explains how to work with source documents in Document Manger.

Adding Source Documents to Projects

You can add source documents to your project when you create a project. You can also add source documents to your project after you create a project. When you add source documents to your project, ePublisher automatically adds the source documents to your project and creates a top-level group in Document Manager that contains your source document. For more information about top-level groups, see “Understanding Source Documents Groups”.

To add a source document to your project

1. On the **Project** menu, click **Add Document**.
2. Browse to the folder that contains the source document you want to add to your project.
3. Select the source document you want to add to your project, and then click **Open**.
4. *If you configured ePublisher to scan source documents when you add source documents to projects*, ePublisher adds the source documents to Document Manager and scans the source documents. For more information about scanning source documents and setting scanning options, see “Scanning Source Documents” and “Setting Scanning Options”.
5. *If you did not configure ePublisher to scan source documents when you add source documents to projects*, ePublisher adds the source documents to Document Manager but does not scan your documents. After ePublisher adds your source documents to Document Manager, scan your source documents. For more information about scanning source documents and setting scanning options, see “Scanning Source Documents” and “Setting Scanning Options”.
6. *If you are adding a FrameMaker book file to your project*, ePublisher adds the FrameMaker book file (.bk or .book files) and the source documents the FrameMaker book file contains (.fm files) to your project. Consider the following points when you add a FrameMaker book file to your project:
 - When you add a FrameMaker book to your project, by default ePublisher creates a group for the FrameMaker book in Document Manager, and any FrameMaker source documents contained within the FrameMaker book are always contained within the group in your project.
 - When you make changes to a FrameMaker book, such as adding or removing source documents from a FrameMaker book file, when you scan the FrameMaker book, ePublisher updates the project with the changes you made to the FrameMaker book file. If you add or remove FrameMaker source documents in a FrameMaker book, ensure you scan the FrameMaker book before you generate output. For more information about scanning source documents and setting scanning options, see “Scanning Source Documents” and “Setting Scanning Options”.
 - *If your FrameMaker book contains front matter files, table of contents files, or index files*, consider the following points:
 - *If you are generating output for a target that uses any output format other than PDF*, by default ePublisher generates output for source document front matter files included in a book, but does not generate output using the table of contents files and index files included in the FrameMaker book. ePublisher instead uses the headings and index entries in the source documents to generate a table of contents and an index for your online output.
 - *If you are generating output for a target that uses PDF as the output format*, by default ePublisher generates the PDF using the front matter, index, and table of contents files included in the FrameMaker book.
 - The Stationery designer may modify these default file processing settings when designing Stationery. If you have target setting modification permissions, you can also customize these settings as

needed. For more information about target setting customization permissions and customizing file processing settings, see “Customizing Target Settings” and “Specifying File Processing Behavior for Front Matter, Index, and Table of Contents Files”. If you do not have target setting customization permissions, instead of adding an Adobe FrameMaker `.book` file that contains front matter, table of contents, and index files, you can instead add the individual Adobe FrameMaker chapter `.fm` files, and then use the individual chapter files to generate output.

After you add source documents to your ePublisher project, ePublisher displays your source documents in Document Manager. You can organize your source documents in Document Manager and perform the following tasks:

- Open and edit source documents from within Document Manager. For more information, see “Opening Source Documents from Document Manager”.
- Relink source documents. For more information, see “Relinking Source Documents”
- Remove source documents from your project. For more information, see “Removing Source Documents from Projects”.
- Create an organizational structure for your online output using groups. For more information, see “Understanding Source Documents Groups” and “Organizing Source Documents Using Groups”.
- Rearrange the source document order in Document Manager. For more information, see “Rearranging Source Documents in Groups”.

Opening Source Documents from Document Manager

If you want to edit the content of your source documents while working with a project, you can open the source documents from Document Manager.

To open a source document from Document Manager

1. In Document Manager, double-click the source document you want to open. ePublisher opens the source document using the content authoring tool you used to create the source document.
2. *If you want to edit the content in your source document*, edit the content using the content authoring tool you used to create the source document.
3. Save the source document.

Scanning Source Documents

This section explains how scanning works, how to scan source documents, and source document scanning options.

Understanding Scanning and Scanning Options

When ePublisher scans your source documents, it reads the style and formatting information, variables, conditions, and marker types in your source documents and then imports this information into your ePublisher project. Once ePublisher imports this information into your project, you can generate output. You can also modify target settings if you have permissions to modify target settings. For more information, see “Generating and Regenerating Output” and “Customizing Target Settings”.

The scanning process can be time-consuming. You can reduce the amount of time it takes ePublisher to scan your documents by scanning only the source documents you select. Scan your source documents when you have made any of the following changes to your source documents:

- Added new content
- Added new style information
- Modified any existing styles
- Added new markers, variables, or conditions
- Modified existing markers, variables, or conditions

ePublisher provides the following options for scanning source documents in Document Manager:

Scan Selected

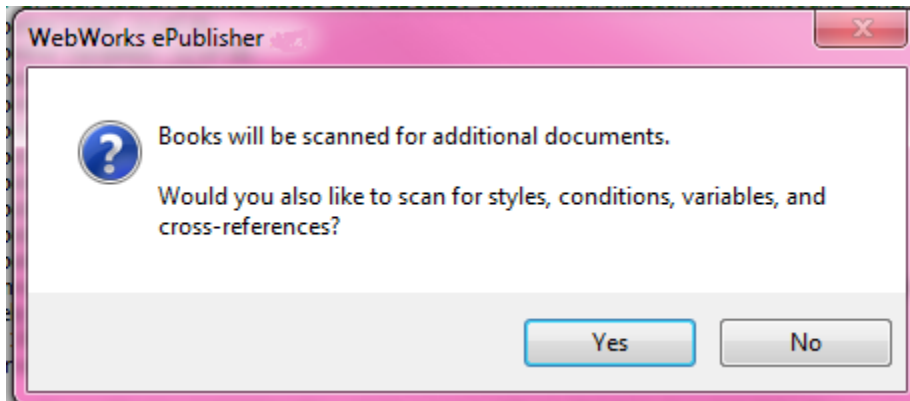
Scans only the selected source document in Document Manager.

Scan All Documents

Scans all of the source documents that you added to your project and that are displayed in Document Manager.

Setting Scanning Options

By default, ePublisher will ask whether or not to scan your documents. When adding a source document a dialog box will appear:



This indicates that files such as FrameMaker .book files will scan for additional .fm files linked from this source. Clicking Yes will scan the individual documents so that the document set styles, for example can be added to the Style Designer (the same goes for most of the document set customizations.)

However, you can specify that you want ePublisher to scan source documents when you add them to your project. For example, you can choose to have ePublisher prompt you to scan the source documents when you add source documents to your project, or you can choose to always have ePublisher scan source documents when you add them to a project. The scanning option you specify will become the default selection for all existing and subsequent projects.

If you choose to never have ePublisher scan source documents, when you add them to a project, you must remember to scan your source documents before you generate output.

To set scanning options

1. On the **Edit** menu, click **Preferences**.
2. On the **General** tab, in the **Scan options** area, select the scan setting you want to specify. For more information about scanning options, click **Help**.
3. Click **OK**.

Scanning Selected Documents

Sometimes you may make a change to content in a single source document. You can scan only the source document you changed. Scanning the selected document updates your project with the new information you specified in the selected source document.

To scan a selected source document

1. In Document Manager, select the source document you want to scan.
2. On the **Project** menu, click **Scan Selected**. ePublisher scans the document you selected in Document Manager.

Scanning All Documents

If you have made multiple changes to content in your source documents, you can scan all of the source documents included in your project at once. Scanning all source documents ensures that ePublisher includes any changes you made to any of the source documents in your project.

To scan all source documents in a project

On the **Project** menu, click **Scan All Documents**. ePublisher scans all of the source documents displayed in Document Manager.

Relinking Source Documents

Sometimes the link between Document Manager and the source document may become broken. For example, moving the source document to another folder location or deleting the source document from a folder may break the link between Document Manager and the source document. When ePublisher detects a broken link between Document Manager and the source document, ePublisher displays a **Broken Link** icon, or red question mark, next to the source document in Document Manager.

To relink a source document

1. In Document Manager, double-click the **Broken Link** icon next to the name of the source document.
2. Browse to the location of the source document.
3. Select the source document, and then click **Open**. ePublisher recreates the link between the source document and Document Manager.

Removing Source Documents from Projects

You can remove source documents from an ePublisher project. Remove source documents from your project when you no longer want to include the content in the source document in your project or in your generated output.

If you are a Stationery designer using ePublisher Designer to design Stationery, when you remove source documents from an ePublisher project, any styles or formats associated with the source document remain in Style Designer. For example, assume that the `UserManualTitle` style is a style that is specific to only one source document in your project. If you remove the source document that contains the `UserManualTitle` style from your project, ePublisher retains the `UserManualTitle` style name and style information in Style Designer. If you want to remove this style from Style Designer, you must manually delete it.

To remove a source document from a project

1. In Document Manager, click the source document you want to remove from your project.
2. On the **Edit** menu, click **Remove**.
3. *If you want to remove an Adobe FrameMaker source document (.fm file) that is a part of an Adobe FrameMaker book (.book or .bk file) you have added to a project*, you cannot remove the Adobe FrameMaker source document from the project using ePublisher. You must remove the Adobe FrameMaker source document from the Adobe FrameMaker book file and then scan the Adobe FrameMaker book file to remove the Adobe FrameMaker source document from your project. For more information about scanning source documents, see “Scanning Source Documents”.
4. Click **Yes** to confirm that you want to remove the source document from your project.

Understanding Source Documents Groups

Groups are containers in Document Manager that hold your source documents and allow you to create an organizational structure for your output. When you first create a new project, ePublisher automatically creates a new group in Document Manager using the project name. You can use ePublisher to create the following types of groups in Document Manager:

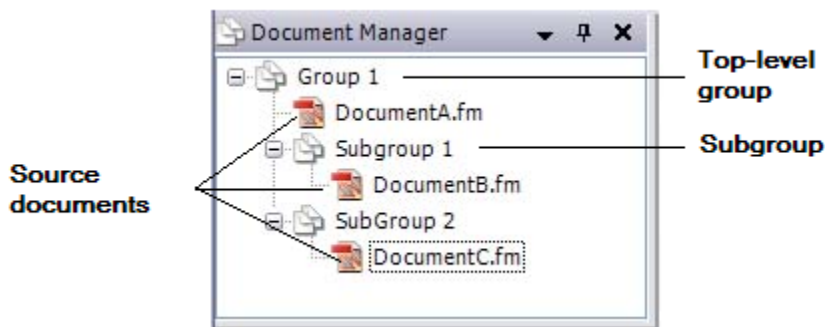
Top-level groups

Contains source documents and subgroups. ePublisher creates an entry-point file for each top-level group in Document Manager. The entry-point file is the file that opens the generated output. All projects must contain at least one top-level group. You can create additional top-level groups to further organize your source documents in Document Manager or if you want to create merged, or multivolume, help. For more information about merged help systems, see “Merging Help Systems (Multivolume Help)”.

Subgroups

Used to organize source documents within top-level groups. Subgroups do not create entry-point files and do not represent an actual volume in a merged help system.

The following figure shows top-level groups and subgroups in Document Manager.



Organizing Source Documents Using Groups

You can perform the following actions with source documents and groups in ePublisher:

- Create top-level groups. For more information, see “Creating Top-Level Groups”.
- Create subgroups. For more information, see “Creating Subgroups”.
- Rename groups. For more information, see “Renaming Groups”.
- Rearrange source documents in groups. For more information, see “Rearranging Source Documents in Groups”.
- Remove groups. For more information, see “Removing Groups”.

Creating Top-Level Groups

By default, ePublisher creates a top-level group based on the name of the project when you add your first source document to your project. There must always be at least one top-level group in Document Manager in order to add source documents to a project. You can create additional top-level groups if you want to further organize your source documents or create merged help systems, or multivolume help. For more information about creating merged help systems, see “Merging Help Systems (Multivolume Help)”.

To create a top-level group

1. On the **Project** menu, click **New Group**. ePublisher creates and displays a new top-level group in Document Manager.
2. Type a name for the new group.
3. Drag the new top-level group to its appropriate position above, below, or between an existing top-level group in Document Manager.

Creating Subgroups

You can create subgroups in Document Manager to organize the source documents in a group. By organizing your source documents into subgroups, you can organize how you want to display your source documents in Document Manager and how you want content to display in your generated output.

To create a subgroup

1. In Document Manager, select the group to which you want to add a subgroup. You can add a subgroup to a top-level group or to an existing subgroup.
2. On the **Project** menu, click **New Group**. ePublisher displays the new group in Document Manager.
3. Type a name for the new group.

Renaming Groups

You can rename existing top-level groups and subgroups in Document Manager. For example, when you create a new project, by default ePublisher creates a new group based on the project name. However, you can change the default name of the group in Document Manager.

To rename a group in Document Manager

1. In Document Manager, click twice on the group you want to rename.
2. Type a new name for the group.
3. Press ENTER or click outside of the typing area to change the name.

Rearranging Source Documents in Groups

Once you have added source documents to your project and placed your source documents into groups within Document Manager, you can rearrange source documents by moving the source documents within the same group or by moving source documents to a new location in a new group.

If you have a FrameMaker book (.bk or .book file) in a group, you can move the FrameMaker book to a different group, but you cannot move an individual FrameMaker document (.fm file) to a group if it is included in the .book file. .fm files that belong to a .book file must remain in the same group as the .book file. If you want to move a .fm file to a different group than the .book file is in, first remove the .fm file from the book, scan the book, and add the .fm file, which is no longer part of the book, to the appropriate book.

To rearrange source documents in groups

1. *If you want to change the order of source documents within a group*, complete the following steps:
 - a. In Document Manager, click the source document you want to move.
 - b. Drag the source document to the desired location within the group.
2. *If you want to move a source document to a different group*, complete the following steps:
 - a. In Document Manager, click the document you want to move.
 - b. Drag the source document to the desired location within the new group.

Removing Groups

If you no longer want to use a group, you can remove the group from Document Manager. When you remove a group from Document Manager, ePublisher removes any source documents associated with the group from your project.

Note: ePublisher does not delete the source documents from your computer. ePublisher only removes the source documents from the project.

To remove a group

1. In Document Manager, select the group you want to remove.
2. On the **Edit** menu, click **Remove**.

Working with Targets

This section explains how to work with targets. For more information about what targets are, see “Understanding Targets”.

Specifying Active Targets

Within a project, you can have multiple targets. The active target is the target currently selected in the project. ePublisher uses the active target when you make modifications to your target settings or generate output.

To specify the active target

On the **Project** menu, select the target next to **Active Target**.

Adding Targets to Projects Based on Stationery

Every project must contain at least one target. Add targets to projects when you need to produce different kinds of output using the same source documents. Each target is associated with one output format, such as WebWorks Help, Microsoft HTML Help, or PDF. If you are generating output based on Stationery using ePublisher Express, the Stationery you use for your project defines the type of output formats you can specify for a target when you add a target to your project. You can only use output formats defined in the Stationery by the Stationery designer when you create targets. If you need to create a target for an output format not included in the Stationery, talk to the Stationery designer about updating the Stationery to include the output format.

For example, assume that you are a writer working at CompanyA, and you need to create web-based help. You have Stationery from a Stationery designer configured to support WebWorks Help, Microsoft HTML Help, and PDF output. In this scenario, you create an ePublisher project based on Stationery from the Stationery designer, and then you create a target called CompanyA WebWorks Help that specifies WebWorks Help as the output format for the target.

Next, assume that your documentation requirements change, and in addition to creating WebWorks Help for CompanyA, you must now also produce Microsoft HTML Help and PDF files for CompanyA using your same source documents. In this scenario, you update your project by adding Microsoft HTML Help and PDF as targets, and your project now contains the following targets:

- CompanyA WebWorks Help
- CompanyA Microsoft HTML Help
- CompanyA PDF Files

To add a target to a project based on Stationery

1. On the **Project** menu, click **Manage Targets**.
2. Click **Add**.
3. In the **Format Type** field, select the output format you want to use for the format target.
4. In the **Target Name** field, type a name for the format target.
5. Click **OK**.

Renaming Targets

You can rename targets. By default, the target name is the same as the output format in ePublisher. However, in some situations, you may want specify a different name for the target. For example, assume that you are a writer working at CompanyA, and you have the requirement to create a web-based help system using your documentation source files. In this scenario, you create an ePublisher project that specifies WebWorks help as your help system and you configure your project settings to use information and branding for CompanyA to create an target called *WebWorks Help*.

Next, assume that your requirements change, and now, based on an OEM agreement your company signed, in addition to creating WebWorks Help for CompanyA, you must use your source files to create WebWorks Help for CompanyB. In this scenario you create a new target in your project called *CompanyB WebWorks Help* and configure settings for this target. However, after configuring settings for the CompanyB WebWorks Help target, you now want to go back and rename your original WebWorks Help output format, and change the name of this output format to *CompanyA WebWorks Help*.

To rename a target

1. On the **Project** menu, click **Manage Targets**.
2. In the **Target Name** field, click the name of the output format you want to rename.
3. Click **Edit**.
4. In the **Target Name** field, type the new name you want to specify.
5. Click **OK**.

Deleting Targets

You can delete targets from a project if you no longer need to produce output for the target.

To delete a target

1. On the **Project** menu, click **Manage Targets**.
2. In the **Target Name** field, click the name of the output format you want to delete.
3. Click **Delete**.
4. Click **OK**.

Working with Projects

This section explains how to work with projects.

Saving Projects

You should periodically save your project to ensure that you do not lose any changes you have made. By saving your project, you ensure that ePublisher stores the information in your project in the project files and all of your project information will be available the next time you open your project.

To save a project

On the **File** menu, click **Save**. ePublisher automatically saves your ePublisher project in a file in the location you specified when you first created the project.

- *If you are saving an ePublisher Express project*, by default ePublisher saves the project file in the `My Documents\ePublisher Express Projects\ProjectName` folder, where *ProjectName* is the name of the project.
- *If you are saving an ePublisher Designer project*, by default ePublisher saves the project file in the `My Documents\ePublisher Designer Projects\ProjectName` folder, where *ProjectName* is the name of the project.

Opening Existing Projects

You can open an existing project using one of the following methods:

- Open the project from within the ePublisher Express or ePublisher Designer user interface.
- Open the project from Windows Explorer by double-clicking the project file in the folder where you saved the project.

By default ePublisher saves project files in the following locations:

- ePublisher saves ePublisher Express project files in the `My Documents\ePublisher Express Projects\ProjectName` folder, where *ProjectName* is the name of the project. ePublisher Express project files use the `.wrp` file extension.
- ePublisher saves ePublisher Designer project files in the `My Documents\ePublisher Designer Projects\ProjectName` folder, where *UserName* is the name of the user account running ePublisher Express and *ProjectName* is the name of the project. ePublisher Designer project files use the `.wep` file extension.

When you open an existing project, ePublisher opens a separate instance of the ePublisher for each project, and each project has its own window. For example, if you have *ProjectA* open, and then you decide to open a project called *ProjectB*, ePublisher opens up a new instance of ePublisher for the new project and you have two ePublisher instances with *ProjectA* and *ProjectB* open concurrently on your computer.

To open an existing project

1. *If you want to open an existing project from within ePublisher Express or ePublisher Designer*, complete the following actions:
 - a. In ePublisher, on the **File** menu, click **Open**.
 - b. Browse to the location of the project file you want to open.
 - c. Select the project file you want to open, and then click **Open**.
2. *If you want to open an existing project using Windows Explorer*, complete the following steps:
 - a. In Windows Explorer, browse to the location of the ePublisher project file you want to open.
 - b. Double-click the ePublisher project file.

Closing Projects

When you finish working with a project, you can close it. When you close the project, ePublisher prompts you to save any changes to your project that you have not already saved.

To close a project

On the **File** menu, click **Exit**.

Synchronizing Projects with Stationery

ePublisher Express projects use Stationery designed in ePublisher Designer by the Stationery designer. From time to time, the Stationery designer may update the Stationery used by your ePublisher Express project. When the Stationery designer updates the Stationery, you must synchronize your ePublisher Express project with the Stationery associated with the project in order to obtain the updates made by the Stationery designer. For more information about Stationery, see “Understanding Stationery”.

When the Stationery designer updates the Stationery an ePublisher Express project uses, ePublisher detects the change the next time you open a project that uses the Stationery, notifies you that the Stationery used by the project has been modified, and prompts you to synchronize your project with the updated Stationery. ePublisher Express prompts you to synchronize your project with its Stationery file under the following conditions:

- ePublisher detects differences between the project manifest file and the Stationery manifest file.
- ePublisher detects modifications to the Stationery file used by the project.

When you synchronize your project with Stationery, you update your project file so that the information in your project file matches the information in the Stationery file and in the Stationery manifest file. Synchronizing the project file with the Stationery file and the manifest file ensures all of the settings and information in the project file match all of the settings and information in the Stationery file. For more information about the Stationery file and the Stationery manifest file, see “Understanding Manifest Files” and “Understanding Stationery Files”.

Based on your ePublisher implementation, after you create an ePublisher Express project using Stationery, you can customize target settings for the targets available in your project if you have appropriate permissions. You can only customize target settings in your ePublisher Express project if you have target setting modification permissions. Any customizations you make to target settings will be overwritten the next time you synchronize your ePublisher Express project with Stationery. For more information, see “Customizing Target Settings”.

ePublisher Express allows you to synchronize your project with its associated Stationery using one of the following methods:

- Automatically synchronize projects with Stationery. For more information, see “Automatically Synchronizing ePublisher Express Projects with Stationery”.
- Manually synchronize projects with Stationery. For more information, see “Manually Synchronizing ePublisher Express Projects with Stationery”.

Understanding Manifest Files

When you create a project based on Stationery in ePublisher Express, ePublisher copies the manifest file used by the Stationery you specify for the project and places a copy of the Stationery manifest file in the project folder for the new ePublisher Express project. The manifest file is a record of all of the files associated with the Stationery file, including all of the files listed in the following project folders:

- `Formats` folder
- `Targets` folder
- `Files` folder

For more information about project folders, see “Understanding the Project Folder Structure”.

Any time the Stationery designer performs one of the following actions in the Stationery `Formats`, `Targets`, or `Files` folder, ePublisher updates the Stationery manifest file:

- Modifies a file in a folder
- Adds a file to a folder
- Removes a file from a folder

When you open an existing ePublisher Express project, ePublisher compares the ePublisher Express project manifest file to manifest file of the Stationery associated with the ePublisher Express project and determines if there are differences between the manifest file.

If the Stationery designer has updated, removed, or added any files to the `Formats`, `Targets`, and `Files` folder in the Stationery since the last time you opened your ePublisher Express project, ePublisher detects these differences and prompts you to synchronize your ePublisher Express project with the Stationery file. When you synchronize your ePublisher Express project with the Stationery file, ePublisher copies the Stationery’s updated manifest file over to your ePublisher Express project file and adds, removes, and updates files in the `Formats`, `Targets`, and `Files` folders for your project as appropriate.

For example, assume that the Stationery designer updated the Stationery you use for one of your projects by adding a new `Page.asp` file. When the Stationery designer makes this change, ePublisher updates the Stationery manifest file with the change. After the Stationery designer makes this change, the next time you open up your ePublisher Express project that uses the changed Stationery, ePublisher Express recognizes that the ePublisher Express project manifest file is different than the Stationery project file and prompts you to synchronize your project to your Stationery file. When you synchronize your project, ePublisher adds the new `Page.asp` file to your project folders.

Understanding Stationery Files

When you open an existing ePublisher Express project, ePublisher Express determines if the Stationery used by the project has been modified by examining the checksum of the Stationery file. A checksum is a value that depends on the contents of a file. ePublisher uses the checksum to determine if a Stationery file has changed. If the checksum of the Stationery file is different than the checksum of the project file, ePublisher Express prompts you to synchronize your project with the Stationery file associated with your project. Any changes to the following settings within the Stationery file will affect the checksum:

- Style and format information
- Conditions
- Variables
- Cross-reference definitions
- Target settings

When to Synchronize

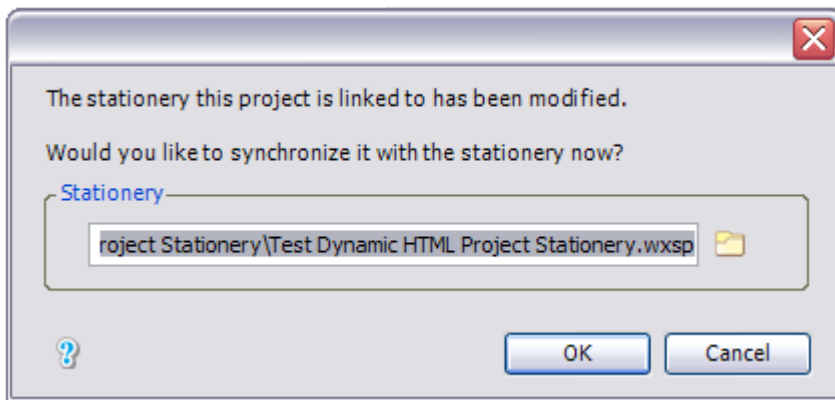
All ePublisher Express projects should be synchronized with Stationery any time the Stationery designer modifies the Stationery. ePublisher Express projects must be synchronized with the Stationery in order for ePublisher to include the changes made by the Stationery designer to the Stationery file in an ePublisher Express project. When you synchronize an ePublisher Express project (.wrp file) with Stationery (.wxsp file), ePublisher updates the information in the ePublisher Express project to match the information in the Stationery file. If you choose not to synchronize, your project will retain its old settings and the information in the project file will not match the information in the Stationery file until you synchronize.

Automatically Synchronizing ePublisher Express Projects with Stationery

When you open an existing project, ePublisher Express automatically detects whether any modifications have been made to the Stationery file. If any changes have been made to the Stationery, ePublisher Express displays a window notifying you that the Stationery has been modified. When this window displays, you can choose to synchronize your project to the modified Stationery file. You can also choose to synchronize your project to new Stationery.

To automatically synchronize an ePublisher Express project with Stationery

1. Open ePublisher Express. If the Stationery designer has modified the Stationery linked to your ePublisher Express project, ePublisher Express displays a window that tells you that the Stationery the ePublisher Express project is linked to has been modified. The window ePublisher displays should be similar to the following window.



2. *If you want to synchronize your ePublisher Express project with the specified Stationery*, click **Yes**.
3. *If you want to synchronize your ePublisher Express project with different Stationery*, complete the following steps:
 - a. Click the folder icon, and then browse to the location of the Stationery with which you want to synchronize your ePublisher Express project.
 - b. Select the Stationery (.wxsp file), and then click **Open**.
 - c. Click **OK** again.
4. *If you do not want to synchronize your ePublisher Express project with Stationery*, click **Cancel**.

Manually Synchronizing ePublisher Express Projects with Stationery

You can manually synchronize your project file with Stationery at any time. When you manually synchronize your project file with Stationery, ePublisher Express prompts you to specify the Stationery with which you want to synchronize your ePublisher Express project. You can synchronize your ePublisher Express project with the Stationery currently associated with your ePublisher Express project, or you can specify that you want your ePublisher Express project to synchronize with different Stationery.

To manually synchronize an ePublisher Express project with Stationery

1. In ePublisher Express, on the **File** menu, click **Synchronize with Stationery**.

Note: You can only synchronize ePublisher Express projects with Stationery. You cannot synchronize ePublisher Designer projects with Stationery, because ePublisher Designer projects are not based on Stationery. ePublisher Designer projects are used to design Stationery.

2. Browse to the location of the Stationery to which you want to synchronize your ePublisher Express project. By default, ePublisher saves Stationery to the following folder:

`My Documents\ePublisher Stationery\ProjectName`, where *ProjectName* is the name of the project used to create the Stationery.

3. Select the Stationery (.wxsp file), and then click **Open**. ePublisher synchronizes the ePublisher Express project with the specified Stationery.

Deleting Projects

Delete a project when you no longer want to use the project to generate output.

To delete a project

1. Open Windows Explorer.
2. Browse to the location of the project folder for the project you want to delete.
 - By default ePublisher saves ePublisher Express project files in the `My Documents\ePublisher Express Projects\ProjectName` folder, where *ProjectName* is the name of the project. ePublisher Express project files use the `.wrp` file extension
 - By default ePublisher saves the ePublisher Designer project files in the `My Documents\ePublisher Designer Projects\ProjectName` folder, where *ProjectName* is the name of the project. ePublisher Designer project files use the `.wep` file extension.
3. Delete the project folder.

When you delete a project, ePublisher continues to display the project on the Start Page until you close and then reopen the ePublisher user interface.

Generating and Regenerating Output

When you generate output in ePublisher, ePublisher creates all of the files specified for the target. ePublisher uses the information in the project source documents and project settings to generate output files. Output files include the following types of files:

- Individual topic page `.html` files (or `.rtf` files if you generate Microsoft WinHelp)
- Image files, such as `.jpg`, `.gif`, and `.png` files
- The entry-point file, which is used to open the generated output
- All files required by the help system if you are generating output for a help system

Understanding Output Generation and Regeneration

In ePublisher, you generate output using either the generate or regenerate option. The generate and regenerate option both create output from your project. However, there are some important differences between the options.

As you make changes to your source documents and your project settings, you need to generate output files in order to see any changes made to the following items:

- Content changes in your source documents
- Changes to project settings
- Changes in the Stationery associated with your ePublisher project

When you *generate* output for a target for the first time, ePublisher creates the output files for the first time. After you generate output files for a target the first time, if you generate output for your target again, you *update* your output files with the changes you made in your source documents and the changes you made to your project settings. Generating, or updating, your output creates output files more quickly than regenerating your output files.

Use the generate option when you have made changes to the following project settings:

- Condition settings
- Variable values
- Cross-reference definitions
- Merge settings
- Target settings
- Project preferences

When you *regenerate* output, ePublisher deletes the `Data` folder from the project folder, creates a new `Data` folder, and *creates* new output files each time. Regenerate your output any time you add new information to your source documents that is not content. Non-content modifications to source documents include adding, removing, or modifying following items:

- Paragraph, character and table styles and formats
- Marker types
- Cross-reference definitions
- Variable values in the source documents
- Condition settings in the source documents

Generating Output

In ePublisher, you can generate output for the following items:

- The entire project, which generates output for all the groups and source documents in your project
- A single group within your project
- An individual source document within your project

Generate output for all of the groups in your project when you are generating the final, completed output or help system, when you are merging output or help systems, or when you are deploying your output.

Generate output for a single group if you have already generated output for the other groups in your project, but you have made some slight modifications to one of the groups. Using ePublisher to generate output for a single group reduces the amount of time it takes ePublisher to generate output for your project. When you generate output for a single group, ePublisher generates output for all of the source documents within the group. If you select a top-level group or a group that contains subgroups, ePublisher generates output for all of the source documents in the group and its subgroups.

Generate output for an individual source document if you have made some slight modifications to a source document and want to preview what your generated output will look like. Selecting an individual source document instead of generating output for the entire group or project reduces the amount of time it takes ePublisher to generate output.

To generate output

1. On the **Project** menu, select the target next to **Active Target** for which you want to generate output.
2. *If you want to generate output for an entire project*, on the **Project** menu, click **Generate All**.
3. *If you want to generate output for a group in your project*, complete the following steps:
 - a. In **Document Manager**, select the group for which you want to generate output.
 - b. On the **Project** menu, click **Generate Selected**.
4. *If you want to generate output for an individual source document in your project*, complete the following steps:
 - a. In **Document Manager**, select the document for which you want to generate output.
 - b. On the **Project** menu, click **Generate Selected**.

Note: Some formats, such as the Wiki-based formats and WebWorks Reverb, must be deployed to a server for proper viewing. WebWorks Reverb does provide a convenience web server that can be used for quick, non-production preview purposes. Refer to *Deploying Output to Output Destinations* for further information.

Regenerating Output

When you regenerate output, ePublisher deletes the `Data` folder from the project folder, creates a new `Data` folder, and generates new output files.

Regenerate your source document any time you have added new information to your source document that is not content, including adding, removing, or modifying the following items:

- Character styles
- Paragraph styles
- Table styles
- Cross-reference definitions
- Variable values within source documents
- Conditions settings within source documents

To regenerate output

1. On the **Project** menu, select the target next to **Active Target** for which you want to regenerate output.
2. On the **Project** menu, click **Regenerate All**.

Generating Output from FrameMaker or Microsoft Word

In 2011.1, ePublisher adds the ability to generate output and reports with your authoring tool via the Transit menu. Output and reports generated via this menu are short-lived. They disappear once the project window is closed. Long lived projects should be created with the classic Express interface.

In your Source Document

1. Go to the **WebWorks** menu
2. Select **ePublisher Express -> Generate Output**
3. Select the Stationery file on which you want the project to be based
4. Click **OK** once you have selected the Stationery and the Target you want to use. Now click **Finish** to generate output. ePublisher provides you a window to view the Output Explorer as well as the generated output

Modifying Help System Title Bars

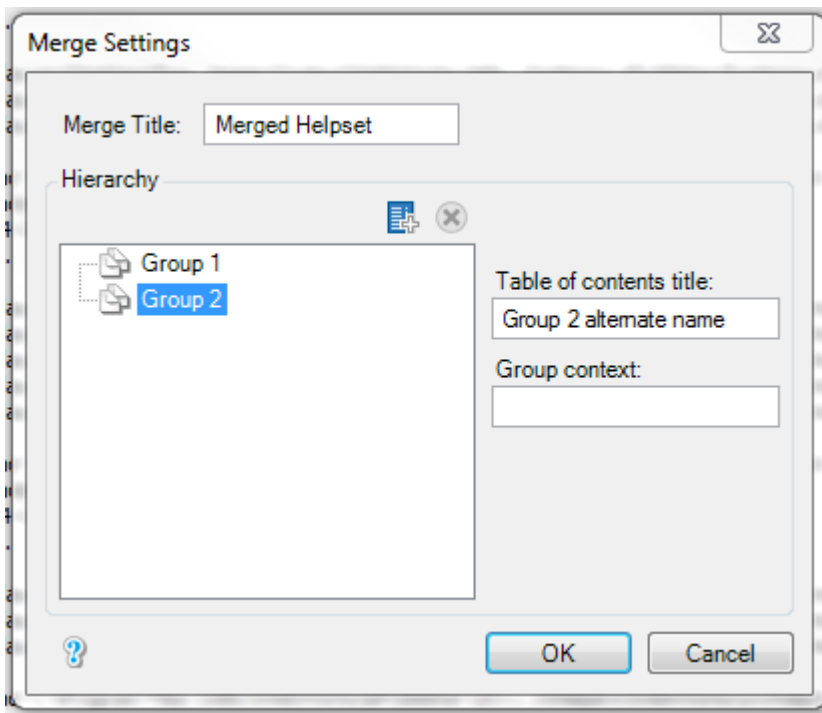
The title bar in your generated help system displays the title you assigned to your project. If you want to specify a different title in the title bar for your generated help system, you can do this in the Merge Settings window if you are generating output for the following help systems:

- Dynamic HTML
- Eclipse Help
- Microsoft HTML Help
- WebWorks Help
- WebWorks Reverb

You cannot use merge settings to modify help system title bars for other output formats.

To modify the title bar of a help system

1. On the **Project** menu, select the target next to **Active Target** for which you want to modify the title bar of a help system.
2. On the **Target** menu, click **Merge Settings**.



3. In the **Merge Title** field, type the title you want to display in the title bar for your generated help system, and then click **OK**.
4. On the **File** menu, click **Save**.
5. Regenerate your output. For more information, see “Regenerating Output”.

Customizing or Removing Splash Page Images in WebWorks Help

The splash page is the first page that displays in the topic frame when a WebWorks Help system first opens. By default, WebWorks Help displays a WebWorks image on the splash page. The Stationery designer may also customize the WebWorks Help splash page in the Stationery used by your project to display a custom image. You can replace the existing splash page image with a different image or you can configure WebWorks Help to display the first topic in the help instead of the splash page image.

Customizing Splash Page Images in WebWorks Help

By default when you generate WebWorks Help, WebWorks Help displays a splash page. The splash page is the first page that displays in the topic frame when the WebWorks Help opens. You can replace the default splash page image with a custom image.

For more information about the project folder, see “Understanding Projects and the Project Folder Structure”.

To replace the splash page image

1. *If you want to override the image for all WebWorks Help targets*, complete the following steps:
 - a. In your project, on the **View** menu, click **Format Override Directory**.
 - b. Create the `WebWorks Help 5.0\Pages\images` folder in your `ProjectName\Formats` folder, where `ProjectName` is the name of your ePublisher project.
2. *If you want to override the image for one WebWorks Help target*, complete the following steps:
 - a. In your project, on the **View** menu, click **Target Override Directory**.
 - b. Create the `WebWorks Help 5.0\Pages\images` folder in your `ProjectName\Targets` folder, where `ProjectName` is the name of your ePublisher project.
3. Copy the `splash.jpg` file from the following folder to the images folder you created within your project folder:
`Program Files\WebWorks\ePublisher\2018.1\Formats\WebWorks Help 5.0\Pages\images`
4. Open the `splash.jpg` file you copied into the images folder and modify it to be the splash page image you want.
5. Save and close the `splash.jpg` file.
6. Regenerate your output and then open WebWorks Help in Output Explorer to verify the change to the splash page image. For more information, see “Regenerating Output” and “Viewing Output in Output Explorer”.

Removing Splash Page Images in WebWorks Help

By default when you generate WebWorks Help, WebWorks Help displays a splash page. However, instead of displaying the splash page, WebWorks Help can display the first topic page instead. If you configure this option, when users open the WebWorks Help system, it displays the first topic.

To remove the splash page image in WebWorks Help

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target** menu, click **Target Settings**.
3. Set **Show first document instead of splash page** to **Enabled**.
4. Click **OK**.
5. Regenerate your output and then open WebWorks Help in Output Explorer to confirm the first topic page displays instead of the splash page. For more information, see “Regenerating Output” and “Viewing Output in Output Explorer”.

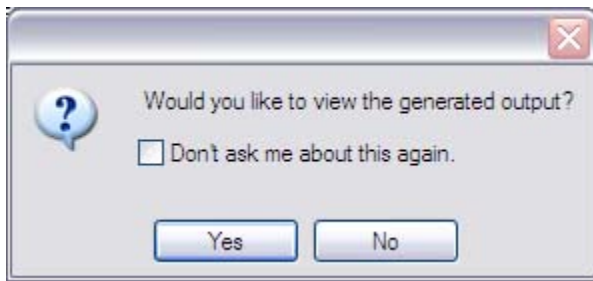
Viewing Output

After you specify project and target settings, generate output for your target to review your changes and verify that the generated output displays and functions properly. You can generate output for all of the source documents and groups in your project, or you can generate output for a single group or source document. You can view your generated output files in one of the following ways:

- View output by automatically opening the generated output. For more information, see “Viewing Output by Automatically Opening Generated Output”.
- View output in Output Explorer. For more information, see “Viewing Output in Output Explorer”.
- View output in the `Output` folder. For more information, see “Viewing Output in the Output Folder”.

Viewing Output by Automatically Opening Generated Output

When you generate or regenerate output for a target, after ePublisher generates output, ePublisher prompts you to view the generated output by displaying the following window:



To view output by automatically opening the output

1. Generate or regenerate output. For more information, see “Generating Output” or “Regenerating Output”.
2. When ePublisher displays a window asking if you would like to view the generated output, perform one of the following actions:
 - *If you want to view the generated output*, click **Yes**.
 - *If you do not want to view the generated output*, click **No**.
 - *If you want ePublisher to automatically open the output each time you generate output and you do not want ePublisher to ask you each time if you want to view the generated output*, select the **Don't ask me about this again** check box, and then click **Yes**.
 - *If you do not want ePublisher to automatically open the output each time you generate output and you do not want ePublisher to ask you each time if you want to view the generated output*, select the **Don't ask me about this again** check box, and then click **No**.

If you select the **Don't ask me about this again** check box and specify that you always want ePublisher to display the generated output or that you never want ePublisher to display the generated output, ePublisher uses the options you specify as the default behavior for automatically displaying output when you generate or regenerate output. If you later want to change the default behavior, you can clear your preferences in the WebWorks ePublisher Preferences window, and then set new preferences the next time you generate or regenerate output. For more information about setting ePublisher preferences, see the *ePublisher Writer Guide*.

Viewing Output in Output Explorer

Output Explorer allows you to view output files from within the ePublisher user interface. Each time you generate output for a group of source documents or for an individual source document, ePublisher displays the generated output files in Output Explorer. The list of files ePublisher displays in Output Explorer is based on if you have a group of source documents selected or if you have an individual source document selected.

If you select a top-level group in Document Manager, ePublisher displays a group folder with the same name as the top-level group in Output Explorer that contains the following items:

- Navigation group. The Navigation group displays the generated entry-point file and printable reports. The entry-point file is the file that opens the generated output.
- Reports group. The Reports group displays any reports associated with the target that ePublisher generated.

If you select a source document in Document Manager, ePublisher displays the source document group with the same name as the source document selected in Document Manager that contains the following items:

- Files group. The Files group contains all of the generated content files and printable reports.
- Images group. The Images group contains images associated with the source document.
- Reports group. The Reports group displays any reports associated with the generated output for the target.

If you select a subgroup in Document Manager, ePublisher does not display any information in the Navigation and Reports groups in Output Explorer, because subgroups do not create a generated entry-point file and do not represent an actual table of contents group in generated output. The entry-point file is the file that opens the generated output.

If you have two or more top-level groups in Document Manager and your output format supports merged help systems, ePublisher creates a Merge Output group in Output Explorer. The Merge Output group contains the entry-point file for the merged help system. For more information about merged help systems, see “Merging Help Systems (Multivolume Help)”.

To view output in Output Explorer

1. ***If Output Explorer is not displayed in the ePublisher user interface***, on the **View** menu, click **Output Explorer**.
2. On the **Project** menu, select the target next to **Active Target** for which you want to view output.
3. ***If you want to view output by opening the entry-point file***, complete the following steps:
 - a. In Output Explorer, select a top-level group.
 - b. Click on the plus sign next to the top-level group to expand the group.
 - c. Click on the plus sign next to the **Navigation** group to expand the group.
 - d. Double-click on the entry-point file to open the generated output.

Output Type Generated	Default Entry-Point File to Double-Click to Open
Dynamic HTML	toc.html
Eclipse Help	View Eclipse Help
Microsoft HTML Help	name.chm

Output Type Generated	Default Entry-Point File to Double-Click to Open
Microsoft WinHelp	<code>name.hlp</code>
Oracle Help	<code>name.jar</code>
Sun JavaHelp	<code>name.jar</code>
WebWorks Help	<code>index.html</code>
WebWorks Reverb	<code>index.html</code>
Wiki - Confluence	You cannot view Confluence generated output by clicking on an entry-point file in Output Explorer. you must deploy the Confluence output to a Confluence server computer before you can view the output. For more information about deploying Confluence output, see “Deploying Output”.
Wiki - MediaWiki	You cannot view MediaWiki generated output by clicking on an entry-point file in Output Explorer. You must deploy the MediaWiki output to a MediaWiki server computer before you can view the output. For more information about deploying MediaWiki output, see “Deploying Output”.
Wiki - MoinMoin	You cannot view MoinMoin Wiki generated output by clicking on an entry-point file in Output Explorer. You must deploy the MoinMoin Wiki output to a MoinMoin Wiki server computer before you can view the output. For more information about deploying MoinMoin Wiki output, see “Deploying Output”.
XML+XSL	<code>toc.xml</code>

4. ***If you generated output for an HTML-based output format and you want to view the individual HTML files generated for a specific document,*** complete the following steps:

Note: By default, ePublisher produces individual HTML files for HTML-based output formats based on the page breaks settings you specify for your project. For more information about specifying page break settings, see “Specifying Page Breaks Settings”.

- a. In Document Manager, select a source document.
- b. In the Output Explorer, click on the plus sign next to the document to expand the group.
- c. Click on the plus sign next to the **Files** group to expand the group.
- d. Double-click on the generated output file to open the file.

5. ***If you generated output for a Wiki-based output format and you want to view the individual .wiki files generated for a specific document,*** you must deploy the generated .wiki files to a Wiki server computer before you can view the individual .wiki files. For more information about deploying Wiki-based output, see “Deploying Output”.

Note: By default, ePublisher produces individual .wiki files for Wiki-based output formats based on the page breaks settings you specify for your project. For more information about specifying page break settings, see “Specifying Page Breaks Settings”.

6. ***If your output format supports merged help systems and you want to view the entry-point file for a merged help system,*** complete the following steps:

- a. In Output Explorer, click on the plus sign next to the **Merged Output** group in the Output Explorer to expand the group.
- b. Double-click on the entry-point file to open the generated output.

Viewing Output in the Output Folder

ePublisher stores generated output pages and images in the `Output` folder. By default, ePublisher creates an `Output` folder in the following location:

- *If you are creating a project using ePublisher Express*, by default ePublisher creates the `Output` folder in the `My Documents\ ePublisher Express Projects \ProjectName` folder, where *ProjectName* is the name of the project.
- *If you are creating a project using ePublisher Designer*, by default ePublisher creates the `Output` folder in the `My Documents\ ePublisher Designer Projects \ProjectName` folder, where *ProjectName* is the name of the project.

The `Output` folder contains individual output folders for each one of your targets. For example, if your project contains targets for WebWorks Help, Microsoft HTML Help, and Dynamic HTML, then there will be three folders, one for each of these targets, in the `Output` folder.

You can view output files for all output formats other than Wiki-based output formats by opening them directly from the `Output` folder. You can only view output files for Wiki-based output formats by deploying the files to a Wiki server computer. For more information about deploying output files to a Wiki server computer, see “Deploying Output”.

You can also view output files for all output formats other than Wiki-based output formats by opening them from the ePublisher user interface. You cannot open files for Wiki-based output formats from the ePublisher user interface. When you open output files from the ePublisher user interface, ePublisher opens the `Output` folder for the active target you are currently working with in ePublisher. For more information about specifying an active target and working with targets, see “Specifying Active Targets” and “Working with Targets”.

To view output in the Output Folder

1. On the **Project** menu, select the target next to **Active Target** for which you want to view output.

Note: You must generate output before you can view output in the `Output` folder. For more information about generating output, see “Generating Output”.

2. On the **View** menu, click **Output Directory**. ePublisher opens Windows Explorer and displays a folder based on the name of your target. This `Output` folder contains the output files ePublisher generated for the active target.

Changing the Location of the Output Folder

When you generate output, ePublisher places the output files into the Output folder. You can modify the location where ePublisher stores your output files.

To change the default location of the Output folder

1. On the **Project** menu, select the target next to **Active Target** for which you want to view output.
2. On the **Target** menu, click **Target Settings**.
3. In the **Generated output location** field, type the path to the folder where you want ePublisher to place the generated output, or click the folder icon to browse to and select a folder.
4. Click **OK**.

Working with Output Log Files

Each time you generate output for a target, ePublisher creates a log file named `generate.log` and writes the following information to the log file:

- Time when output generation began
- Actions and commands ePublisher performed, such as processing, creating and copying files
- Pipelines processed by ePublisher
- Any messages, warnings, or errors generated by ePublisher when ePublisher generated output for the target
- Time when output generation ended
- Total amount of time it took ePublisher to generate the output

To work with output log files for a target

1. If you want to view the log file for a target from within the ePublisher user interface, complete the following steps:
 - a. On the **Project** menu, select the target next to **Active Target** for which you want to view log files.
 - b. On the **View** menu, click **Log Window**.
2. If you want to save the log file as a `.txt` file, complete the following steps:
 - a. Click the **Save** button, located in the upper-right corner of the Log Window.
 - b. Specify a name for the log file and the location where you want to save the log file, and then click **Save**.
3. If you want to view the log file for a target using Windows Explorer, in Windows Explorer browse to one of the following locations:
 - **If you are using ePublisher Express**, browse to the `ProjectName\Logs\TargetName` folder, where `ProjectName` is the name of the project and `TargetName` is the name of the target for which you generated output. By default ePublisher saves project files for ePublisher Express projects in the `My Documents\ePublisher Express Projects\ProjectName` folder, where `ProjectName` is the name of the project.
 - **If you are using ePublisher Designer**, browse to the `ProjectName\Logs\TargetName` folder, where `ProjectName` is the name of the project and `TargetName` is the name of the target for which you generated output. By default ePublisher saves project files for ePublisher Designer projects in the `My Documents\ePublisher Designer Projects\ProjectName` folder, where `ProjectName` is the name of the project.

Validating Output Using Reports

After you generate output, you can validate your output using ePublisher reports. ePublisher reports contain information about how ePublisher processed items in your source documents when ePublisher generated output. Reports also allow you to identify any problems that occurred when ePublisher generated output. If reports display notifications, such as messages, warnings, or errors, you can correct the items in your source documents that caused the error. You can then generate output again and then review the reports again to verify that any issues have been addressed as needed.

ePublisher provides the following types of reports:

- Accessibility reports. For more information, see “Understanding Accessibility Reports”.
- Baggage Files reports. For more informations, see “Understanding Baggage Files Reports”.
- Conditions reports. For more information, see “Understanding Conditions Reports”.
- Filenames reports. For more information, see “Understanding Filenames Reports”.
- Links reports. For more information, see “Understanding Links Reports”.
- Styles reports. For more information, see “Understanding Styles Reports”.
- Topics reports. For more information, see “Understanding Topics Reports”.
- Images reports. For more information, see “Understanding Images Reports”.
- Printable reports. For more information, see “Understanding Printable Reports”.

For more information about configuring and generating reports, see “Configuring Reports” and “Generating Reports”.

Understanding Accessibility Reports

You can use markers in your source documents to create accessible online content. For more information about using markers to creating accessible content, see the *ePublisher Writer Guide*.

You can use Accessibility reports to validate that the online content you generate using ePublisher meets your accessibility requirements. Accessibility reports provide notifications on the following items when ePublisher generates output:

- Images without alternative text
- Image maps without alternative text
- Images without long descriptions
- Tables without summaries

Configure the notifications you want ePublisher to generate for Accessibility report settings before you generate Accessibility reports. For more information about configuring Accessibility report settings, see “Configuring Reports”. For more information about generating Accessibility reports, see “Generating Reports”.

Understanding Baggage Files Reports

You can generate baggage files by adding links to HTML or PDF files in your file system, in your source documents. For more information about baggage files, see “Understanding Baggage Files” and “Specifying Baggage Files Settings”.

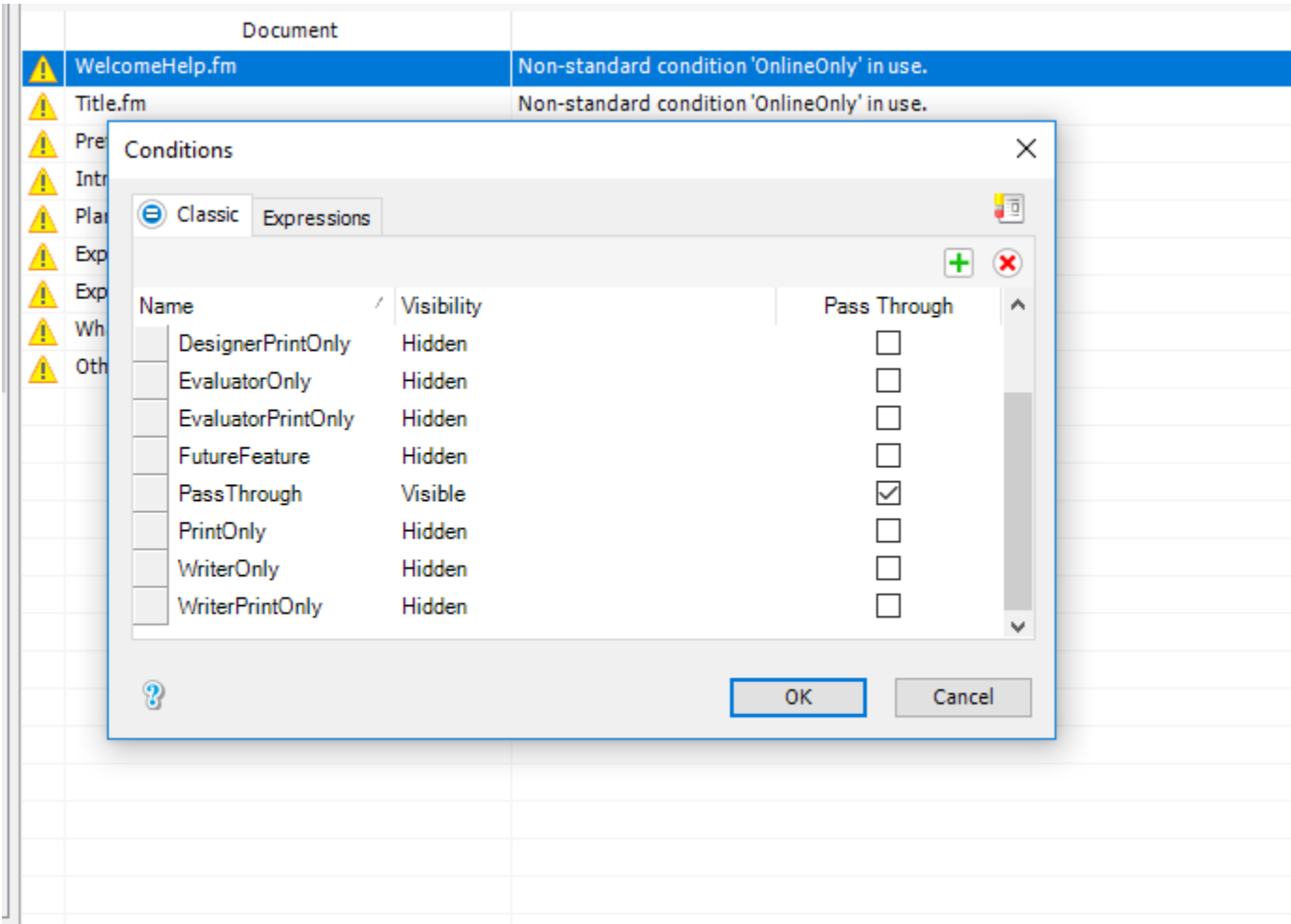
You can use Baggage Files reports to obtain information about how the baggage files are going to be shown in your Search Results. Baggage Files reports provide notifications on the following items when ePublisher generates output:

- Baggage files without summary
- Baggage files without title

Configure the notifications you want ePublisher to generate for Baggage Files report settings before you generate Baggage Files reports. For more information about configuring Baggage Files report settings, see “Configuring Reports”. For more information about generating Baggage Files reports, see “Generating Reports”.

Understanding Conditions Reports

If you add new conditions after ePublisher already scanned the document, those new conditions won't be picked up by ePublisher. To warn you about that, ePublisher has a Conditions Report where you can see all those details, and re-scan the document so the new conditions can be picked up or add it manually yourself using the UI:



Understanding Filenames Reports

You can specify names for output files using Filename markers. For more information about specifying output file names using Filename markers, see the *ePublisher Writer Guide*.

You can use Filenames reports to validate that ePublisher named your output files correctly using the Filename markers you inserted in your source documents. The Filenames report displays the name of the Filename marker you inserted into your source document and the name of the output file ePublisher generated based on the Filename marker. The Filenames report also provides notifications on the following items when ePublisher generates output:

- The files ePublisher created that correspond to the Filename markers you inserted into your source documents
- If ePublisher ignored a Filename marker when generating output
- If duplicate Filename markers exist in the source documents used by your project to generate output

Configure the notifications you want ePublisher to generate for Filenames report settings before you generate Filenames reports. For more information about configuring Filename report settings, see “Configuring Reports”. For more information about generating Filename reports, see “Generating Reports”.

Understanding Links Reports

You can use Links reports to verify that the links you specify to items in your source documents resolve and that ePublisher processed links in your source documents to the item referenced by the link correctly. Links reports provide notifications on the following items:

- Baggage files
- External URLs
- Unresolved links to items in other documents
- Unresolved links to missing source document
- Unresolved links to missing files
- Unresolved link within source documents document
- Unsupported baggage files
- Unsupported external URLs
- Unsupported group to group links

For the definition of a baggage file see “Understanding Baggage Files”.

Configure the notification you want ePublisher to generate for Links report settings before you generate Links reports. For more information about configuring Links report settings, see “Configuring Reports”. For more information about generating Links reports, see “Generating Reports”.

Understanding Styles Reports

Styles reports allow you to verify that your source documents conform to the styles and formatting defined in the Stationery by the Stationery designer. The Styles report notifies you about the following items when ePublisher generates reports:

- Any non-standard styles used in your source documents
- Any style overrides used in your source documents

A non-standard style is any style that exists in your source document but is not defined in the stationery file used by your project. For example, if you add a new style to your source document called `BodyIndent 4`, but your stationery designer has not updated the stationery file to include the `BodyIndent 4` style, the Styles report notifies you that there is a non-standard style used in the source document.

A style override is any modification you made to the original style definition for a particular instance of a style. For example, if you have applied the `Body` paragraph style to a paragraph in your source document, and you then apply the `Bold` character style to the paragraph, the `Body` paragraph style has a style override.

If your source document contains any non-standard styles or style overrides, ePublisher will process your source documents when you generate output using the non-standard styles and style overrides you applied in your source documents.

Configure the notifications you want ePublisher to generate for Styles report settings before you generate Styles reports. For more information about configuring Styles report settings, see “Configuring Reports”. For more information about generating Styles reports, see “Generating Reports”.

Understanding Topics Reports

Context-sensitive help topics require that you have TopicAlias markers inserted in your source documents. ePublisher generates context-sensitive help topics based on the topic IDs you specify for each TopicAlias marker you insert in your source documents. Each time ePublisher detects a TopicAlias marker in a source document, ePublisher generates a context-sensitive help topic based on the topic ID. For more information about creating context-sensitive help topics, see the *ePublisher Writer Guide*.

You can use the Topics Report to verify that context-sensitive help topics have been created for each topic ID specified in your source document. The Topics Report lists the topic ID and the topic file created for each topic ID.

Configure the notifications you want ePublisher to generate for Topics report settings before you generate Topics reports. For more information about configuring Topics report settings, see “Configuring Reports”. For more information about generating Topics reports, see “Generating Reports”.

Understanding Images Reports

Image reports enable you to verify the integrity and appearance of ePublisher manage images. Users are notified any time a source image is missing or when an image occurs in a problematic structure, such as images within tables in the ePUB format.

Understanding Printable Reports

With the new ePublisher you can see your reports in two different ways: from the UI and in your browser (Printable Reports). You can find them right after your usual reports.

The screenshot shows the ePublisher interface. On the left is the 'Output Explorer' pane, which lists various reports under the 'Reports' folder. The 'Printable' sub-folder is expanded, showing 'Styles Report (Printable)', 'Links Report (Printable)', 'Filenames Report (Printable)', 'Topics Report (Printable)', 'Images Report (Printable)', 'Conditions Report (Printable)', and 'BaggageFiles Report (Printable)'. The 'Filenames Report (Printable)' is selected. On the right is a browser window titled 'Anamany' with the address bar showing a file path. The browser displays the 'Filenames Report' as a table with three columns: 'Severity', 'Document', 'Description', and 'Links'. The table contains three rows of warnings, each with a yellow warning icon in the 'Severity' column. The 'Document' column contains links to 'Creating Context Sensitive Topics.fm'. The 'Description' column contains text about filename markers being ignored or processed. The 'Links' column contains 'Source' and 'Output' links.

Severity	Document	Description	Links
⚠	Creating Context Sensitive Topics.fm	Filename marker 'context-sensitive help topics:creating' has been ignored.	Source Output
⚠	Creating Context Sensitive Topics.fm	Filename marker 'IDH_CreatingContextSensitiveHelpTopicsFrame' has been ignored.	Source Output
⚠	Creating Context Sensitive Topics.fm	Filename marker 'context-sensitive help topics:instructions' has been processed as 'context-sensitive help topics_3ainstructions' for generated file 'context-sensitive help topics_3ainstructions.html'.	Source Output

Configuring Reports

When you use reports to validate your output, you must specify the type of notification that you want to display when ePublisher detects issues or performs actions while generating output using your source documents. When ePublisher generates output, ePublisher generates notifications under the following conditions:

- When ePublisher cannot properly process elements
- When ePublisher encounters missing information

For example, ePublisher can generate a notification when it detects a potential error in your source documents when you generate output, such as an unresolved cross reference. ePublisher can also generate a notification when it performs a specific action using elements contained in your source documents, such as when ePublisher generates an output file using a filename you specified using a Filename marker.

You can specify the following values for report options when you generate output:

Ignore

Specify this value if you do not want ePublisher to report any issues it identifies in the report. For example, specify this value you do not want the Styles report to report any style overrides.

Message

Specify this value if you want to receive a message when ePublisher completes or fails to complete an action. For example, if you are not concerned if your source document uses non-standard styles, but you would like to see where non-standard styles are used in your source documents, specify this value.

Warning

Specify this value if you want to receive a warning when ePublisher completes or files to complete an action. For example, if you want to be warned when ePublisher detects non-standard styles in your source documents, specify this value.

Error

Specify this value if you want the report to display an error when ePublisher completes or files to complete an action. For example, if you want to receive an error notification when ePublisher detects unresolved cross-references in your source documents, specify this value.

To configure report notification settings

1. On the Project menu, select the target next to **Active Target** for which you want to configure report notification settings.
2. On the **Target menu**, click **Target Settings**.
3. *If you want to specify Accessibility report notification settings*, in the **Accessibility Report** area, specify a value for each Accessibility report notification setting you want to configure. For more information about each setting, click **Help**.
4. *If you want to specify Baggage Files report notification settings*, in the **Baggage Files Report** area, specify a value for each Baggage File report notification setting you want to configure. For more information about each setting, click **Help**.
5. *If you want to specify Conditions report notification settings*, in the **Conditions Report** area, specify a value for each Condition report notification setting you want to configure. For more information about each setting, click **Help**.

6. *If you want to specify Filenames report notification settings*, in the **Filenames Report** area, specify a value for each Filename report notification setting you want to configure. For more information about each setting, click **Help**.
7. *If you want to specify Links report notification settings*, in the **Links Report** area, specify a value for each Link report notification setting you want to configure. For more information about each setting, click **Help**.
8. *If you want to specify Styles report notification settings*, in the **Styles Report** area, specify a value for each Style report notification setting you want to configure. For more information about each setting, click **Help**.
9. *If you want to specify Topics report notification settings*, in the **Topics Report** area, specify a value for each Topic report notification setting you want to configure. For more information about each setting, click **Help**.
10. *If you want to specify Images report notification settings*, in the **Images Report** area, specify a value for each Image report notification setting you want to configure. For more information about each setting, click **Help**.

Generating Reports

You can generate reports for source documents and baggage files by selecting the group or source document that you want to generate reports for in Document Manager. Before you generate reports, configure notification settings for each report you want to generate. For more information about configuring report notification settings, see “Configuring Reports”.

To generate a report

1. In Document Manager, select the group or source document for which you want to generate a report.
1. *If you want to generate all reports for the selected item*, on the **Project** menu, click **Generate Reports > All**.
2. *If you want to generate Accessibility reports for the selected item*, on the **Project** menu, click **Generate Reports > Accessibility Report**.
3. *If you want to generate Baggage Files reports for the selected item*, on the **Project** menu, click **Generate Reports > Baggage Files Report**.
4. *If you want to generate Conditions reports for the selected item*, on the **Project** menu, click **Generate Reports > Conditions Report**.
5. *If you want to generate Filename reports for the selected item*, on the **Project** menu, click **Generate Reports > Filenames Report**.
6. *If you want to generate Links reports for the selected item*, on the **Project** menu, click **Generate Reports > Links Report**.
7. *If you want to generate Styles reports for the selected item*, on the **Project** menu, click **Generate Reports > Styles Report**.
8. *If you want to generate Topics reports for the selected item*, on the **Project** menu, click **Generate Reports > Topics Report**.
9. *If you want to generate Images reports for the selected item*, on the **Project** menu, click **Generate Reports > Images Report**.

Understanding Report Messages

The following tables provide descriptions for report messages.

Accessibility Report Messages

The following table lists messages in Accessibility reports.

Message	Definition
Table is missing a table summary.	The table does not contain a table summary. Insert a table summary marker within the table.
Image link '{0}' is missing alternate text.	The hotspot does not have alternate text. Insert an image area alternate text marker in a text frame within the image.
Image is missing alternate text.	The image does not have alternate text. Insert an image alternate text marker in a text frame within the image.
Image is missing a long description	The image does not have a long description. Insert an image long description marker in a text frame within the image.

Baggage Files Report Messages

The following table lists messages in Baggage Files reports.

Message	Definition
Title missing for '{0}'.	<p>The baggage file doesn't have a title defined.</p> <p>If it's an <code>HTML</code> baggage file insert a <code><title></code> tag in the <code><head></code> tag of the <code>HTML</code>. Or add the <code>@title</code> attribute to that file entry in your baggage list info file.</p> <p>If it's a <code>PDF</code> baggage file add the <code>@title</code> attribute to that file entry in your baggage list info file.</p>
Summary missing for '{0}'.	<p>The baggage file doesn't have a summary defined.</p> <p>If it's an <code>HTML</code> baggage file you can do one of these:</p> <ul style="list-style-type: none">• insert a <code><meta></code> tag in the <code><head></code> tag of the <code>HTML</code> with <code>@name='summary'</code> and <code>@content</code> with the summary you want to define,• create any kind of tag inside the <code><body></code> tag that accepts the attribute <code>@class</code> with <code>@class='summary'</code> and then place your summary as the content of the element,• insert a <code><meta></code> tag in the <code><head></code> tag of the <code>HTML</code> with <code>@name='description'</code> and <code>@content</code> with the summary you want to define. <p>Or add the <code>@summary</code> attribute to that file entry in your baggage list info file.</p> <p>If it's a <code>PDF</code> baggage file add the <code>@summary</code> attribute to that file entry in your baggage list info file.</p>

Filename Report Messages

The following table lists messages in Filename reports.

Message	Definition
File '[NAME]' has been processed as a baggage file.	Any files not contained within your project are processed as baggage files.
Filename marker '[NAME]' has been used for generated file '[FILE PATH]'.	A file has been generated using a filename marker. This alerts you that the name of the file has been changed.
Filename marker '[NAME]' has been ignored.	The filename marker has been ignored because it is either uses a duplicate name or it has not been inserted at a heading that splits.
Filename marker '[NAME]' has been processed as '[NAME]' for generated file '[NAME]'.	The filename marker has not been used; instead, the file has been renamed to the filename indicated.

Links Report Messages

The following table list messages in Links reports.

Message	Definition
Unresolved link to target '[NAME]' in document '[NAME]'.	There is an unresolved cross-reference in the document. The destination target either does not exist or cannot be found.
Unresolved link from document '[NAME]' to target '[NAME]' in document '[NAME]'.	There is an unresolved cross-reference from a document to a location in another document. The destination target cannot be found.
Unresolved link from document '[NAME]' to document '[NAME]'.	There is an unresolved link from one document to another document. It cannot find the referenced document.
Unresolved link from document '[NAME]' to missing file '[NAME]'.	There is an unresolved link from a document to an external file. A file refers to any file that is not part of the ePublisher project or is not of the same type as your source document (for example, .jpeg, .gif, .tif)
Unresolved link from document '[NAME]' to document '[NAME]'. Output format does not support group to group linking.	There is an unresolved cross-reference from one document to another document because the output format your project is using does not support linking from one top-level group to another.
Unresolved link from document '[NAME]' to file '[NAME]'. Output format does not support baggage files.	There is an unresolved cross-reference from the document to a file because the output format your project is using does not support baggage files. Files refer to any file that is not part of the project.
External URL link '[NAME]' is not supported.	The output format does not support external links.

Styles Report Messages

The following table lists messages in Styles reports.

Message	Definition
Encountered [STYLE TYPE] style name '[NAME]' in your source file that is not defined in ePublisher.	The style name is not defined in ePublisher.
Encountered text with [STYLE TYPE] style name '[NAME]' that has modified style properties in your source file.	There is a style override. Style overrides refer to attributes that are defined within the style.
Encountered style properties not associated with a named style in your source file.	There is a style override. For example, the character style <i>bold</i> has been modified in one instance of its use.

Topics Report Messages

The following table lists messages in Topics reports.

Message	Definition
Topic '[NAME]' resolves to the file '[FILE PATH]'.	A topic page has been created for the topic alias marker.
Topic '[NAME]' is duplicated in the file '[FILENAME]'	A duplicate topic alias has been created in that file.

Images Report Messages

The following table lists messages in Images reports.

Message	Definition
Missing by-reference source files	An image referenced by the source document is missing.
Images in table cells	Image occurs inside a table cell (problematic for certain ePUB readers)

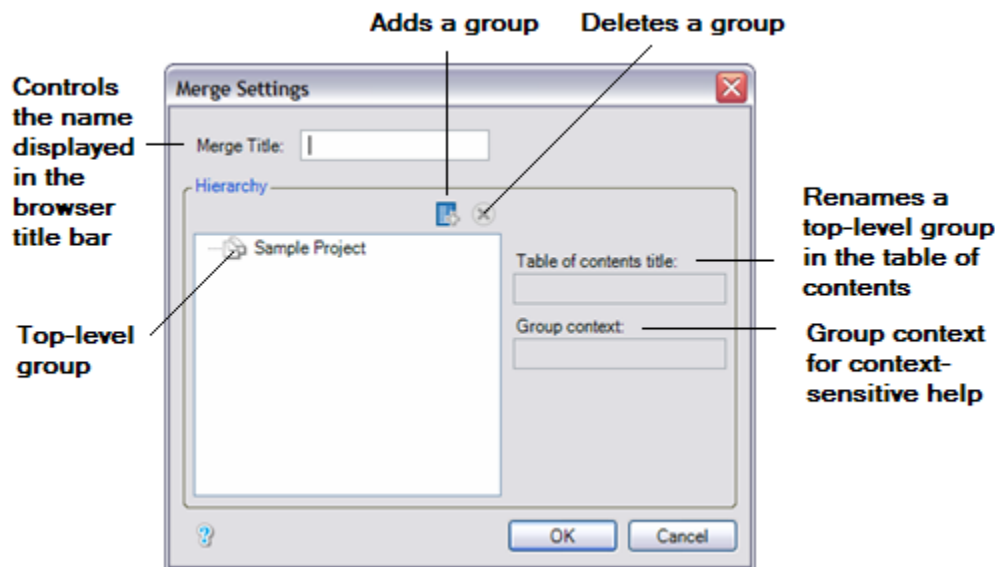
Merging Help Systems (Multivolume Help)

Merged help, which is sometimes also referred to as multivolume help, is a help system with a single set of files created from output from multiple groups from within a project. Merged help takes the table of contents, index, and search data from each top-level group entry-point file and combines this information to create a single, consolidated help system. You can use ePublisher to create merged help systems for the following output formats:

- Eclipse Help
- Microsoft HTML Help
- WebWorks Help
- WebWorks Reverb

If you have created several top-level groups in Document Manager for your project, by default ePublisher generates its own help system with its own entry-point file when you generate output for your project. The entry-point file is the file that opens the help system. ePublisher places the merged help system in the Merged Output group in Output Explorer.

You must have at least two top-level groups in Document Manager to create merged help. By default, ePublisher uses the organizational structure specified in Document Manager to create the merged, or multivolume help system. If you want to organize and group your top-level groups using a different name than the group name specified in the Document Manager, or if you want to use a different hierarchy in your merged help system than the hierarchy you currently have specified for your project in Document Manager, you can do this using merge settings. The following figure shows the Merge Settings window.



ePublisher names the merged help system based on the name of your target. For example, if you generate output for a target named CompanyA WebWorks Help, ePublisher creates an entry-point file for the merged help system named CompanyA WebWorks Help and displays this name in the title bar when users open the merged help system.

ePublisher also creates individual help systems for each top-level group in Document Manager and names these individual help systems based on the names of the top-level groups in Document Manager. For example, if you have three top-level groups in Document Manager named FeatureA, FeatureB, and FeatureC and you are generating output for a target called CompanyA WebWorks Help, ePublisher creates FeatureA, FeatureB, and

FeatureC help systems as well as a merged help system named CompanyA WebWorks Help that merges the table of contents, index, and search data from each top-level group into a single, consolidated help system. These top-level groups also display in the table of contents in your merged help system.

You can use ePublisher merge settings to perform the following actions:

- Specify a different name than the target name for the title displayed in the title bar of the merged help system
- Specify a different name for subgroups in your generated output than the names used in Document Manager
- Organize and group your top-level groups in a merged help system into a different hierarchy than the hierarchy used in Document Manager.

To merge help systems

1. Create the top-level groups you want to use in your merged help system in Document Manager. For more information about creating top-level groups, see “Creating Top-Level Groups”.
2. On the **Project** menu, select the target next to **Active Target** for which you want to create a merged help system.
3. On the **Target** menu, click **Merge Settings**.
4. *If you want to specify a name other than your target name for the merged help system*, in the **Merge Title** field, type in the name you would like to display in the title bar of your merged help system.
5. *If you want to specify a different name for each top-level group in the table of contents for your generated output*, complete the following steps for each top-level group you want to rename in your generated output:
 - a. In the **Hierarchy** area, select the name of the top-level group for which you want to specify a different name in the generated output.
 - b. In the **Table of contents title** field, type the name you want to display for the group in the generated output.
6. *If you want to reorganize the table of contents in your merged help system*, select and then drag and drop any of the top-level groups to a new position.
7. *If you want to create a new custom group for your merged help system that includes some of your existing top-level groups from Document Manager*, complete the following steps:
 - a. Click the **Add** button. The **Add** button in the Merge Settings window is an icon of a blue page with a plus (+) character ePublisher adds a new group called `Untitled Topic` to your table of contents hierarchy.
 - b. Click on the `Untitled Topic` group in the Merge Settings window and rename it.
 - c. Select and then drag and drop the top-level groups you want to include in the new group into the new group.
8. *If you want to delete a custom group you previously created that contains top-level groups*, complete the following steps:

Note: You can only remove groups that you have manually added to your merged help system hierarchy. You cannot remove groups ePublisher creates by default based on the top-level groups in Document Manager.

 - a. Select the group you want to remove.

- b. Click the **Delete** button.
9. *If you are generating merged, or multivolume WebWorks Help or WebWorks Reverb that includes context-sensitive help*, in the **Group context** field, specify the help context for each top-level group to use.

Note: In WebWorks Help and WebWorks Reverb, you need to include the context and the TopicAlias value in the help call to display the correct help topic. For more information, see “Using Context-Sensitive Help in WebWorks Help”.
10. Click **OK**.
11. Generate your output. For more information, see “Generating Output”.
12. Open the merged help system by completing one of the following steps:

Note: ePublisher creates the entry-point file using the name of the selected target. If you want to change the name of the entry-point file for the merged help system, rename your target. For more information about renaming your target, see “Renaming Targets”.

 - a. On the **View** menu, click **Output Explorer**.
 - b. Under the Merge Output group in the Output Explorer, double-click on the entry-point file for the merged help system to open the merged help system.

Note: Ensure you click under the Merge Output group in Output Explorer. You must click under the Merge Output group in Output Explorer in order to view the merged output. If you click under one of the other groups, you will only see the output generated for the specific group selected.
13. Review the merged help system you created based on the merge settings you specified and confirm that your merged help system displays using the help system name and table of contents group hierarchy that you want.

Deploying Output

This section explains how you can use ePublisher to deploy output to multiple locations, such as to folders on a network, to a Web server, or to a Wiki.

Understanding Output Deployment

By default ePublisher places output files in the following location on your local computer:

- *If you are creating a project using ePublisher Express*, by default ePublisher creates the Output folder in the `My Documents\ePublisher Express Projects\ProjectName` folder, where *ProjectName* is the name of the project.
- *If you are creating a project using ePublisher Designer*, by default ePublisher creates the Output folder in the `My Documents\ePublisher Designer Projects\ProjectName` folder, where *ProjectName* is the name of the project.

If you would like to deploy your output files to another location in addition to this default location after ePublisher generates output, such as a folder on a network, you can deploy your output to one or more output destinations using ePublisher. The **output destination** is the location where you would like to deploy your generated output files. In ePublisher, the output destination consists of the following components:

- Output name
- Output destination location

To deploy your output, you must perform the following steps:

1. Create one or more output destinations. For more information, see “Creating Output Destinations”.
2. Specify an output destination for each target. For more information, see “Specifying Output Destinations for Targets”.
3. Deploy output to output destinations. For more information, see “Deploying Output to Output Destinations”.

Creating Output Destinations

Before you can deploy your output, you must create output destinations. You can specify one output destination or multiple output destinations. Specify multiple output destinations when you want to deploy your output to multiple locations. For example, assume that you place your generated output to a web server computer or to a Wiki, and you use both a staging server and a production server. You can create one output destination in ePublisher for the staging server, and another output destination in ePublisher for the production server.

Output destinations are not project or target specific. When you define output destinations in ePublisher, ePublisher saves the output destinations you define and allows you to use the output destinations you specify across multiple ePublisher projects and targets.

When you deploy output to an output destination, ensure you specify a descriptive name for the output destination. When you work with output destination, you can only see the name of the output destination. You will not be able to see the actual path you specified to the output destination. Type a descriptive name for the output destination that allows you to easily identify each output destination you specify.

For example, if you are deploying WebWorks Help output for a product to both a staging server and a production server, type `Production Server ProductA WebWorks Help` for the first output destination. When you create your second output destination, type `Staging Server ProductA WebWorks Help` for the second output destination.

If you are deploying Wiki - MediaWiki or Wiki - MoinMoin output, review Wiki output format requirements before creating an output destination on a Wiki. For more information, see “Wiki - MediaWiki” and “Wiki - MoinMoin”.

To create an output destination

1. On the **Target** menu, click **Target Settings**.
2. Click **Add deploy target**.
3. *If your target is not based on a Wiki output format*, complete the following steps:
 - a. Click **Add > Folder**.
 - b. In the **Name** field, type a descriptive name for the output destination.
 - c. In the **Directory** field, type the path to the folder you want to specify as the output destination, or click the folder icon and then browse to and select the folder where you would like to deploy your output.
 - d. Click **OK**.
4. *If your target is based on a Wiki output format*, such as Wiki - MediaWiki or Wiki - MoinMoin, complete the following steps:
 - a. Click **Add > Wiki - Type**
where **Type** is the Wiki type output format on which your target it based.
For example, click **Add > Wiki - MediaWiki** or **Add > Wiki - MoinMoin**
 - b. In the **Name** field, type a descriptive name for the output destination.
 - c. Click **Edit**.
 - d. In the **Wiki Location** field, specify the URL root on the Wiki where you want to deploy the output.

- e. *If you want to deploy output for a target that uses the Wiki - MoinMoin output format and you want to deploy different versions of MoinMoin Wiki content to the same MoinMoin Wiki*, specify a parent page where you want to deploy the MoinMoin Wiki content. The parent page is the URL page path where you want to deploy the Wiki output.
- f. *If the Wiki to which you are deploying uses authentication*, select the **User name and password** check box, and then specify the user name and password for a user account with permissions on the Wiki. ePublisher encrypts this information before storing it.
- g. Click **Test**. ePublisher checks to see if the user credentials you specify have appropriate permissions on the Wiki by requesting an authentication token from the Wiki.
- h. Click **OK**.

After you create an output destination, you must specify which target is associated with the output destination before you can deploy output. For more information, see “Specifying Output Destinations for Targets” and “Deploying Output to Output Destinations”.

Specifying Output Destinations for Targets

After you create an output destination, you must associate the output destination with an target before you can deploy output.

To specify an output destination for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify an output destination.
2. On the **Target** menu, click **Target Settings**.
3. In the **Deploy to** field, select an output destination.

Note: You must create an output destination before you can specify an output destination for a target. For more information about creating output destinations, see “Creating Output Destinations”.

4. Click **OK**.

After you specify an output destination for a target, you can generate output and deploy the output to the output destination.

Deploying Output to Output Destinations

After you create output destinations, specify output destinations for targets and generate output, you can use the **Deploy** command in ePublisher to copy your output files and place them into the locations you specified as output destinations. When you deploy output, ePublisher copies the target output files and places the output files in the location you specified as the output destination. For more information about creating output destinations and specifying output destinations for targets, see “Creating Output Destinations” and “Specifying Output Destinations for Targets”.

If you are deploying Wiki - MediaWiki or Wiki - MoinMoin output, review Wiki output format requirements and ensure the output destination you specify has been configured appropriately in order to support deployment of Wiki - MediaWiki or Wiki - MoinMoin output format. For more information about Wiki output format requirements, see “Wiki - MediaWiki” and “Wiki - MoinMoin”.

To deploy output to an output destination

1. On the **Project** menu, select the target next to **Active Target** for which you want to deploy output.
2. On the **Target** menu, click **Deploy**. ePublisher deploys the output files to the specified output location.

Customizing Target Settings

Based on your ePublisher implementation, after you create a project using Stationery, you can customize target settings for the targets available in your project if you have appropriate permissions. You can only customize target settings in a project if you have target setting modification permissions.

If you are using ePublisher Designer, you have target setting modification permissions. If you are using ePublisher Express, you may or may not have target setting modification permissions. When you install ePublisher Express, you must select the **Allow users to modify Target Settings and Properties** check box in order to have permissions to modify the target settings for the targets available in your project. If you do not select this check box during installation, you will not be able to customize target settings in projects. However, you can enable target setting modification permissions after you install ePublisher Express if needed. For more information, see “Working with Contract IDs”.

If you have permissions to modify the target settings in projects, you can customize the following target settings for most output formats:

Note: If you are using ePublisher Express, any customizations you make to target settings will be overwritten the next time you synchronize your ePublisher Express project with Stationery. For more information, see “Synchronizing Projects with Stationery”.

- Accessibility settings. For more information, see “Specifying Accessibility Settings”.
- Baggage Files settings. For more information, see “Specifying Baggage Files Settings”.
- Company information. For more information, see “Specifying Company Information”.
- File processing behavior for front matter, index files, and table of contents files. For more information, see “Specifying File Processing Behavior for Front Matter, Index, and Table of Contents Files”.
- When to create new pages. For more information, see “Specifying Page Breaks Settings”.
- How you want to name your page files and image files when generating output. For more information, see “Specifying Page, Image, and Table File Naming Patterns”.
- Index settings. For more information, see “Specifying Index Settings”.
- How links to files or external URLs display in browser windows. For more information, see “Specifying How Links to Files or External URLs Display in Browser Windows”.
- Character encoding settings for targets. For more information, see “Specifying Character Encoding for Targets”.
- Language used by targets. For more information, see “Specifying the Language Used by Targets”.
- PDF generation settings. For more information, see “Specifying PDF Generation Settings”.
- Table of contents settings. For more information, see “Specifying Table of Contents Settings”.
- Report settings. For more information, see “Specifying Report Settings”.
- Output format-specific settings, such as settings specific to the WebWorks Help output format or the Microsoft HTML Help output format. For more information, see “Specifying Output Format-Specific Settings”.
- Variable settings. For more information, see “Customizing Variable Settings in Projects”.

- Condition settings. For more information, see “Customizing Condition Settings in Projects”.
- Cross-reference settings. For more information, see “Customizing Cross-Reference Settings in Projects”.

After you make any customizations to the target settings for the targets available in your project, generate output so that you can review your changes and verify that the generated output displays and functions properly. You can generate output for all the source documents and groups in your project, or you can generate output for a single group or source document. For more information about generating output, see “Generating Output”.

Specifying Accessibility Settings

In ePublisher, accessibility refers to how users with disabilities access electronic information and how writers and producers of online content produce accessible output that can function with assistive devices used by individuals with disabilities. Creators of online content, such as writers who produce online content and help systems and others who are responsible for producing accessible help, or Section 508 compliant content, must follow certain guidelines established by the W3C and the U.S. government. If you are responsible for producing accessible online content, you must provide alternate text and descriptions for all images and image maps and summaries for all tables included in the online content. Ensure you specify this information when you prepare your source documents for output generation. For more information, see the *ePublisher Writer Guide*.

To specify accessibility settings for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target** menu, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **Accessibility**, specify the appropriate values for the Accessibility settings. For more information about Accessibility settings and values, click **Help**.
4. Click **OK**.

Specifying Baggage Files Settings

In addition to your source files content inside HTML and PDF files can be indexed by enabling Baggage File settings. In ePublisher you can access to the Baggage File settings only if your target is WebWorks Reverb (also Reverb 2). There are several settings that allows you to customize the way Reverb handles the baggage files which will only take effect when you have ON the **Client-side Search** option in **WebWorks Reverb**, see “Client-side Search”.

You can specify the following settings:

- Baggage files info list
- Copy baggage file dependents
- Create standalone group

Note: If your Output Format is **Reverb 2** this setting not supported.

- Index baggage files
- Index external links
- Standalone group name

Note: If your Output Format is **Reverb 2** this setting not supported.

We will explain each setting briefly below.

Baggage files info list

The `baggage files info list` allows you to specify additional baggage files and external URLs that you want to be included in the search index results and/or packaged with the output.

Note: Currently, creating baggage files in this way requires that the **Create standalone group** Target setting be **Enabled**.

Note: External URLs cannot be packaged with the help, however, they can be included as part of the search index. Furthermore, external URLs must enable the **Create standalone group** Target setting in order to be indexed.

If your baggage files or external URLs do not have adequate title and summary values defined, then you can also use the `baggage files info list` to provide these values. These values are used for displaying search results.

If you have external URL links or baggage files in your source content that you do not want included in the search index, then use the attribute `@noindex`.

When specifying the path for baggage files, you can either use an absolute path or a path relative to the `baggage files info list` file.

You can specify the location and filename of the `baggage files info list` using the Target Settings dialog (explained below). By default the filename is called: `baggage_list.xml` and is available to override in the **Advanced > Manage Format/Target Customizations** menu. In addition to renaming this file, you can also specify an absolute path or a relative path from the project file directory. Furthermore, you can use a variable for getting the directory location of the first document in the project. Using this variable called: `$FirstDocDir;`, you can locate the `baggage files info list` file in this directory. This is a useful way to use your stationary with multiple projects.

To use this variable you need to specify it first in the Target Setting value like this:

```
$FirstDocDir;/baggage_list.xml.
```

Note: If you change the path of the `baggage files info list` target setting, then even if you have overridden this file, the overridden file will be ignored and the file specified will be used instead. However, if you just change the base filename, then ePublisher will look for this file as if it were an override.

To get the path of the Baggage Files info list file we follow these steps

1. ***If you change the default value or just change the name (without specifying a path)*** Reverb tries to get the file from the Targets folder first and then from the Formats folder. The file will be located in the Transforms folder, so you can easily do an override of it if you want to keep the default name, otherwise you can add the new file there with the name you defined in the **Target Settings**.
2. ***If you define an Absolute or Relative path*** Reverb will calculate the *relative path* relative to the project file and will take the *absolute path* as is.

If you don't specify a title and/or a summary for a baggage file we will try to do it for you.

In case of an HTML file for getting a title (if you didn't define one in a baggage list file) we will search for:

1. A `<title>` tag defined in the `<head>` section of the HTML.
2. The base name without the extension of the HTML file.

In case of a PDF file for getting a title (if you didn't define one in a baggage list file) we will get the base name without the extension of the PDF file.

In case of an HTML file for getting a summary (if you didn't define one in a baggage list file) we will search for:

1. The attribute `@content` in the `<meta>` tag defined in the `<head>` section of the HTML with attribute `@name="summary"`.
2. All the text inside the first tag, contained in the `<body>` tag, with attribute `@class="summary"`.
3. The attribute `@content` in the `<meta>` tag defined in the `<head>` section of the HTML with attribute `@name="description"`.
4. All the text inside the first `<p>` tag, contained in the `<body>` tag.

In case of a PDF file for getting a summary (if you didn't define one in a baggage list file) we will get the first 300 letters from the content of the PDF file.

The attribute `@noindex` accepts 2 values: `true|false`, or you can even not define this attribute at all, and it will take the value `false` by default. If you define `@noindex="true"` it means Reverb won't index that file.

The attribute `@path` is for specifying the path to the file (relative or absolute) or the external URL. It should be an existing path to an HTML page or PDF, either local or in the web.

Note: Any text you write in this file, if it contains a reserved character you'll have to change it to use the entity corresponding. For example, instead of "&" use "&".

The following code will show you how to structure a baggage list file, as well as some examples for the entries:

```
<?xml version="1.0" encoding="UTF-8" ?>
<Files version="1.0" xmlns="urn:WebWorks-Baggage-List-Schema">
<File path="http://example.com/" noindex="true"/>
<File path="https://example.com/myfavoritepage" title="My favorite page" summary="Favorite
pages can make your day better"/>
<File path="Source-Docs\some_pdf.pdf" title="Some PDF title" summary="Some PDF summary"/>
<File path="C:\Documents\another_pdf.pdf" title="Another PDF title" summary="Another PDF
summary"/>
...
</Files>
```

Copy baggage file dependents

If you have this setting **Enabled** in your **Target Settings**, all the dependents of your HTML baggage files will be copied to the `baggage` folder inside the corresponding group in the output folder. These mean the final user will be able to open the HTML baggage file and it will look pretty similar to the original one. Right now we support the dependences corresponding to these tags:

- `<link>` tag inside the `<head>` tag that it's not an external URL or starts with "javascript:"
- `<script>` tag that it's not an external URL or starts with "javascript:"
- `` tag that it's not an external URL or starts with "javascript:"
- `<input>` tag that it's not an external URL or starts with "javascript:"
- `<iframe>` tag that it's not an external URL or starts with "javascript:"
- `<video>` tag that it's not an external URL or starts with "javascript:"
- `<audio>` tag that it's not an external URL or starts with "javascript:"
- `<object>` tag inside the `<body>` tag that it's not an external URL or starts with "javascript:"

Create standalone group

The behavior by default when you have baggage files linked in your source documents is to copy them to your Output folder in the corresponding group folder. Reverb creates a `baggage` folder inside the group folder and it copies there all your baggage files and its dependents (if you have **Copy baggage file dependents** -> **Enabled**), and also copies creates the corresponding `bpair.js` files containing the pairs in each baggage file to the `pairs` folder inside that group. Additionally, in your group index file (with format `NameOfTheGroup_sx.js` in the output target folder), Reverb adds the indexed words for every baggage files with its relevance.

Note: If your Output Format is **Reverb 2** Standalone groups are not supported and this section does not apply.

The problem with the default behavior is that if you use the same baggage file across different groups, you'll have it duplicated in your Output folder, because it's going to be copied to every group folder and furthermore you are going to have duplicate results in your Search Results pointing to the different groups where that file lives.

But if you have the **Create standalone group** setting **Enabled** in your **Target Settings**, Reverb instead of copying to every group the corresponding baggage files, it will create an “imaginary” parallel group to the other groups that it will only contain 2 folders: `baggage` and `pairs`, containing all related with the baggage files across all groups. Also it will create an index file, with all the indexed words in each baggage file (named `StandaloneGroupName_sx.js`, using as `StandaloneGroupName` the one defined in **Standalone group name**) in the output target folder. Now you will have an independent index for all your baggage files and all the information related with them in the same place (a common folder).

Note: If you have this setting **Enabled** and the **Index baggage files** setting -> **Enabled** all the entries in the `baggage files info list` representing baggage files that are not in your source documents will be indexed as well. So this is a way to index files not contained in your source. Also, if you have the **Index external links** setting **Enabled** combined with this setting, Reverb will index those external links from the `baggage files info list`.

Index baggage files

If you have this setting **Enabled** in your **Target Settings**, Reverb will index all the baggage files allowing them to show up in Search Results. This as mentioned before requires you have them linked in your source documents or in your Baggage files info list (this requires **Create standalone group** -> **Enabled**). You can override this behavior on a file by file basis by specifying the attribute `@noindex="true"` in your Baggage files info list.

In order to handle most any type of HTML file Reverb uses Tidy (tool for cleaning up HTML files) for creating a well-formed XHTML temporary copy of the files, which are valid XML files that ePublisher can read.

Index external links

If you have this setting **Enabled** in your **Target Settings**, Reverb will index all the external links you have in your source documents and in your Baggage files info list (if you have **Create standalone group** -> **Enabled**). That means you'll have in your Search Results links to your external URLs if they match with the searched phrase.

The Reverb format downloads the file to the Data directory and then uses Tidy (tool for cleaning up HTML files) for creating an XHTML copy of the files, which are valid XML files that ePublisher can read.

Note: If your URL needs to execute some JavaScript code to get the content of the page, Reverb won't be able to index the dynamic content of the page. To simulate the actual content that Reverb will index at a particular URL, temporarily disable JavaScript in your browser and visit that link.

Standalone group name

If you have the **Create standalone group** setting **Enabled** in your **Target Settings**, this will be the name the Standalone Group will take, see “Create standalone group”.

Note: If your Output Format is **Reverb 2** Standalone groups are not supported and this section does not apply.

To specify baggage files settings for a target (only for Reverb targets):

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target menu**, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **Baggage Files**, specify the appropriate values for the Baggage Files settings. For more information about Accessibility settings and values, click **Help**.
4. Click **OK**.

Specifying Company Information

You can add your company's contact information to each generated output page. By default, ePublisher displays the company contact information on the bottom and/or top of your output pages. Where the company information displays depends on what the Stationery designer specified in the Stationery file.

You can specify the following company information:

- Company email address
- Company fax number
- Company logo image
- Company name
- Company phone number
- Company web page

To specify company information for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target** menu, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see "Customizing Target Settings".
3. Under **Company Information**, specify the appropriate values for the company information settings. For more information about the company information settings and values, click **Help**.
4. Click **OK**.

Specifying File Processing Behavior for Front Matter, Index, and Table of Contents Files

You can specify file processing behavior for front matter, index files, and table of contents files. For example, you can specify whether or not you want to generate output for front matter included in your source documents.

To specify file processing behavior for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target menu**, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **File Processing**, specify the appropriate values for file processing settings. For more information about the file processing settings and values, click **Help**.
4. Click **OK**.

Specifying Page Breaks Settings

When ePublisher processes source documents, it creates new topic pages based on settings specified by the Stationery designer in the Stationery. However, you can modify how you would like ePublisher to handle the page breaks.

To specify page break settings for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target** menu, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **Files**, in the **Page break handling** field, select the value you want to specify. For more information about the values, click **Help**.
4. Click **OK**.

Specifying Page, Image, and Table File Naming Patterns

You can specify page, image, and table file naming patterns that you want ePublisher to use when generating output.

For example, you can specify if you would like to include the following items in page, image, and table file names when generating output:

- Target name
- Name of the group in Document Manager that contains the topic
- Page heading text or title

You can use image naming patterns to specify names for embedded image output files. However, if you insert your images by reference in Adobe FrameMaker or use the **Link to File** or **Insert and Link** option in the Insert Picture window in Microsoft Word, ePublisher preserves the original file names.

You can only specify table file naming patterns for Wiki - MoinMoin output. When ePublisher generates Wiki - MoinMoin Wiki, it creates a separate file for each table in a topic.

Note: You can also use Filename markers to specify page and image output file names.

For more information about using markers to specify output file names, see “Using Markers to Define File Names” and “Specifying Table File Naming Patterns for Wiki - MoinMoin Output”.

To specify page, image, and table file naming patterns for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target** menu, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **Files**, specify the appropriate values for the page, image, and file naming patterns you want to use. For more information about file settings and values, click **Help**.

Note: You can only specify table file naming patterns for targets that use the Wiki - MoinMoin output format.

4. Click **OK**.

Specifying Index Settings

In ePublisher, you can specify if you want to generate an index for your help system. If you choose to generate an index for your help system, you must have index markers in your source documents. For more information about creating index markers in your source documents, see the *ePublisher Writer Guide*.

To specify index settings for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target** menu, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **Index**, specify the appropriate value for each index setting. For more information about the index settings and values, click **Help**.
4. Click **OK**.

Specifying How Links to Files or External URLs Display in Browser Windows

ePublisher allows you to specify how you want links that open baggage files or links that open external URLs displayed in your output. For the definition of a baggage file see “Understanding Baggage Files”.

To specify link settings for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target menu**, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **Links**, specify the appropriate value for each links setting. For more information about the links settings and values, click **Help**.
4. Click **OK**.

Specifying Character Encoding for Targets

In ePublisher, encoding refers to the character encoding method used to convert bytes into characters. Programs use encoding when they display HTML documents. Documents in English and most other Western European languages typically use the widely supported character encoding UTF-8. If you are producing output localized for other languages, such as Japanese, Korean, Simplified Chinese, Traditional Chinese, Greek, Turkish, or Eastern European, Cyrillic, or Baltic languages, you must specify the correct encoding for each target for which you generate output.

Ensure the encoding you specify when you generate your output matches the encoding used in the environment where your output will be posted. For example, if your output will be posted on a web server, the encoding you specify when you generate your output should match the encoding used on the web server. If your output and the computer or web server hosting your output do not use the same character encoding method, some characters may not display correctly when users view your output.

To specify character encoding for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target** menu, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **Locale**, specify the appropriate value for the **Encoding** setting. For more information about the encoding setting values, click **Help**.
4. Click **OK**.

Specifying the Language Used by Targets

In ePublisher, locale refers to the language used when displaying output for a target. If you produce localized output, specify the correct language for each target in your ePublisher project.

To specify the language to use for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target menu**, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **Locale**, specify the appropriate value for the **Locale** setting. For more information about the locale setting values, click **Help**.
4. Click **OK**.

Specifying PDF Generation Settings

ePublisher can generate PDFs for each source document, for each top-level group in your project, or for each source document and each top-level group in your project.

Note: *If you are generating WebWorks help and you want to display a PDF button in your WebWorks Help system*, see “Specifying Output Format-Specific Settings”.

To specify PDF generation settings for a target

1. On the **Project** menu, select the output format next to **Active Target** for which you want to specify settings.
2. On the **Target** menu, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **PDF**, specify the appropriate values for the PDF settings. For more information about PDF settings and values, click **Help**.
4. Click **OK**.

Specifying Table of Contents Settings

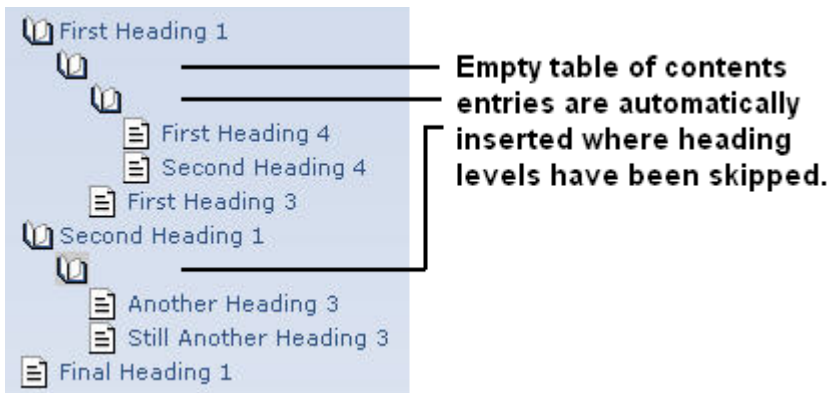
ePublisher allows you to specify whether you want to generate a table of contents, the file name you want to specify for your table of contents file, and how you want your table of contents to display in your generated output.

ePublisher provides table of contents settings to help you address how you want your table of contents to display. By default, ePublisher uses the table of contents levels specified in the project or in the Stationery file to create a table of contents for your help system based on the heading levels in your source documents. However, if you have source documents where writers skipped heading levels, you can specify how you want ePublisher to display skipped headings in the table of contents.

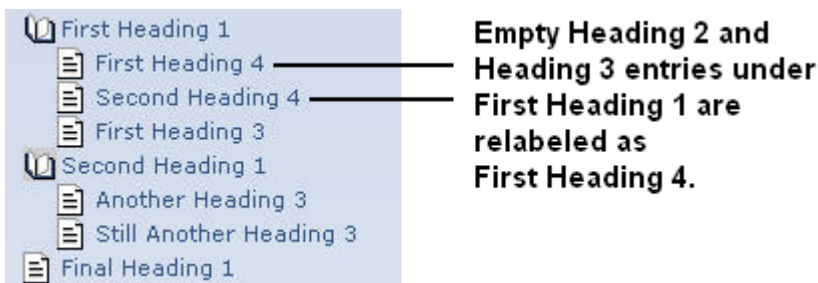
For example, assume that you have an ePublisher project that uses a Stationery file that specifies Heading 1, Heading 2, and Heading 3 as levels in the output table of contents. Then assume that in the source document, you skipped several Heading 2 levels. ePublisher displays an empty table of contents icon, similar to the following figure, in the location of the skipped Heading 2 levels unless you specify how you want to manage skipped heading levels in the generated table of contents.

You can specify the following behavior for table of contents where writers skipped headings:

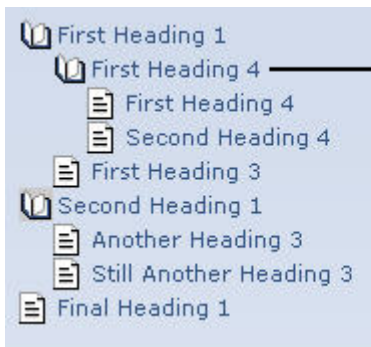
- *If you want ePublisher to automatically insert empty table of contents entries for skipped heading levels,* select the **Don't collapse** value. The following figure shows a table of contents with this option selected.



- *If you want ePublisher to automatically insert labeled entries for skipped heading levels,* select the **Re-label** value. ePublisher displays the heading text from the table of contents entry below the current entry as the table of contents label. The following figure shows a table of contents with this option selected.



- *If you want ePublisher to automatically remove empty table of contents entries and move the heading that follows an empty table of contents entry up a level to replace the skipped table of contents level,* select the **Smart collapse** value. The following figure shows a table of content with this option selected.



Extra table of contents entry automatically inserted to ensure that Heading 4 is nested deeper than Heading 3.

- *If you want ePublisher to remove all skipped heading levels and table of contents entries and place all table of contents headings at the same level, regardless of the table of contents level specified in the Stationery, select the **Fully collapse** value. The following figure shows a table of contents with this option selected.*

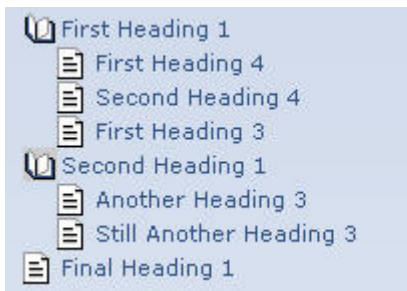


Table of contents is fully collapsed, which allows the Heading 4 and Heading 3 to appear at the same level.

To specify table of contents settings for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify settings.
2. On the **Target** menu, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under **Table of contents**, specify the appropriate values for the table of contents settings. For more information about table of contents settings and values, click **Help**.
4. Click **OK**.

Specifying Report Settings

You can use reports to identify problems in your source documents. If an ePubublisher report detects problems in your source document, ePubublisher displays a notification alert in the report. You can specify which types of settings you want to use to validate your generated output and the type of notification you want to receive if ePubublisher detects an issue when validating your output. For more information about using reports to validate your output and the different types of notifications you can receive, see “Validating Output Using Reports”.

Specifying Output Format-Specific Settings

You can specify output format-specific settings for the following output formats:

- eBook - ePUB 2.0
- Eclipse Help
- Microsoft HTML Help
- Microsoft WinHelp
- Oracle Help
- PDF
- PDF - XSL-FO
- Sun JavaHelp
- WebWorks Help
- WebWorks Reverb
- Wiki - Confluence
- Wiki - MediaWiki
- Wiki - MoinMoin

You must have the target that uses the output format selected in your project before you can see the output format-specific settings in the window. For example, to see WebWorks Help output format-specific settings in the window, you must have a target that uses the WebWorks Help output format selected as your active target. If you have a target that uses the Microsoft HTML Help output format selected as your active target, you will not be able to see WebWorks Help output format-specific settings in the window. You will only be able to see Microsoft HTML Help output format-specific settings.

To specify output format-specific settings for a target

1. On the **Project** menu, select the target next to **Active Target** for which you want to specify output format-specific settings.
2. On the **Target** menu, click **Target Settings**. You must have target modification permissions to modify target settings. For more information, see “Customizing Target Settings”.
3. Under the name of the output format, specify the appropriate values for each output format-specific setting. For more information about output format-specific settings and values, click **Help**.
4. Click **OK**.

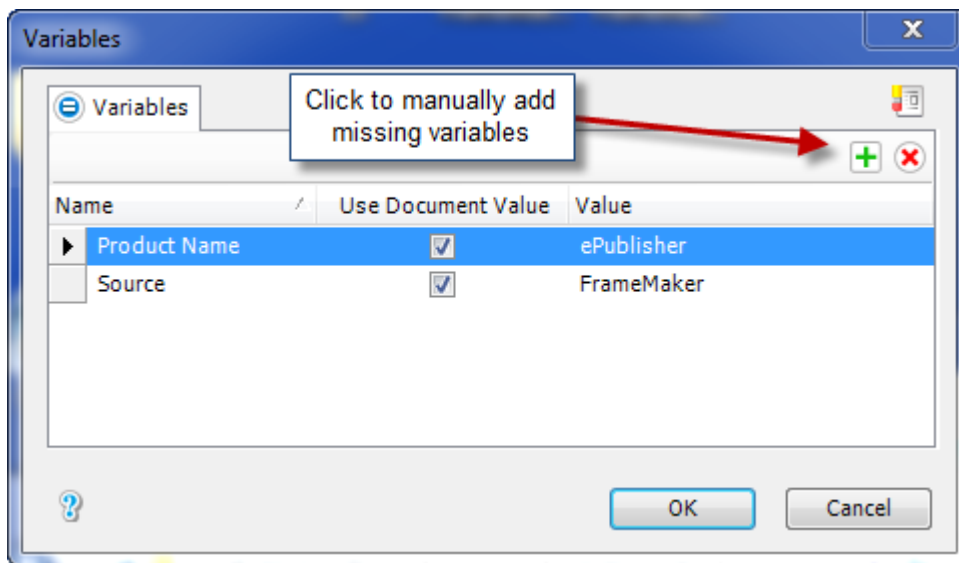
Customizing Variable Settings in Projects

In a project, you can use the variable values defined in your source document. You can also change the value of any variable in your source document in a project. Changing the value of a variable in a project does not change or affect the value of the variable in your source document. You can use the value of the variable you defined in your project when you generate output. Before you can work with variables in projects, you must insert variables in your source documents. For more information about variables and inserting variables and conditions in your source documents, see the *ePublisher Writer Guide*.

To customize a variable in a project

1. On the **Project** menu, select the target next to **Active Target** for which you want to customize variable settings.
2. On the **Target** menu, click **Variables**. You must have target modification permissions to modify variable settings for a target. For more information, see “Customizing Target Settings”.

Note: For Microsoft Word documents only, if you use variables that are built-in DocProperty types such as *Author* or *Company*, then you will need to manually add these variables into the project or stationery as they are not detected when scanned by ePublisher Designer. However, once added into either the stationery or project then they will be available for customization from that point forward.



3. In the **Name** column, find the variable you want to modify.
4. *If you want the your ePublisher project to use the variable value defined in your source document*, click in the **Value** field for the variable, and then select **Use document value** from the drop-down list.
5. *If you want to change the variable value ePublisher uses when generating output*, click in the **Value** field for the variable, and then type in a new value for the variable.
6. Click **OK**.
7. Generate your output. For more information, see “Generating Output”.
8. Review your output and confirm that variables display appropriately in your generated output. For more information, see “Viewing Output”.

Customizing Condition Settings in Projects

In a project, you can use the conditions defined in your source document to control the visibility of content to which you have applied conditions. You can also change the visibility specified for any condition in a project. Changing the visibility specified for any condition in a project does not change the visibility specified for the condition in your source documents. Before you can work with conditions in projects, you must apply conditions to content in your source documents. For more information about conditions and applying conditions in your source documents, see the *ePublisher Writer Guide*.

To customize a condition in a project

1. On the **Project** menu, select the target next to **Active Target** for which you want to customize condition settings.
2. On the **Target** menu, click **Conditions**. You must have target modification permissions to modify condition settings for a target. For more information, see “Customizing Target Settings”.
3. In the **Name** column, find the condition for which you want condition to set the value.
4. Specify the appropriate value for the condition. For more information about condition values, click **Help**.
5. Click **OK**.
6. Generate your output. For more information, see “Generating Output”.
7. Review your output and confirm that conditionalized content displays appropriately in your generated output. For more information, see “Viewing Output”.

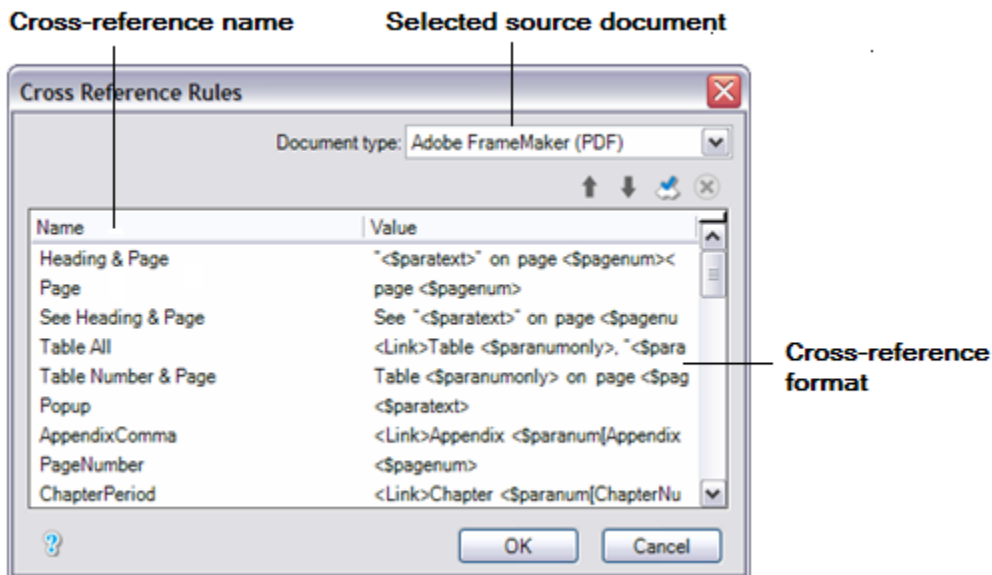
Customizing Cross-Reference Settings in Projects

Cross-references help users access related information quickly in printed and online content. When you convert your source documents to online help, if you have cross-references in your source documents, ePublisher automatically converts all cross-references to hypertext links. Typically, cross-references used for printed materials have a different format than cross-references used for online help. For example, cross-references in printed content typically include page numbers, while cross-references in online help typically do not include page numbers, because page numbers are out of context in online help.

In ePublisher, you use the Cross-Reference Rules window to add, edit, or delete cross-reference formats for your project. A cross-reference format is a combination of text and code that defines how you want your cross-reference to display. For example, your source documents may display the following cross-reference format: “Modifying Cross-Reference Formats on page xxx”, where xxx is the page number where the topic “Modifying Cross-Reference Formats” begins. However, you may modify the cross-reference format in ePublisher so that when you generate online content, the “Modifying Cross-Reference Formats” topic displays as a hyperlink without a page number, such as Modifying Cross-Reference Formats in Projects.

ePublisher obtains the cross-reference formats and values in the Cross-Reference Rules window from your source documents. You can modify cross-reference formats in ePublisher. For more information about cross-reference building blocks or codes, see your content authoring tool documentation.

The following figure shows the Cross-Reference Rules window in ePublisher.



Modifying Cross-Reference Formats in Projects

Modify cross-reference formats when you want cross-references in your online content to use a different format than your printed content.

To modify a cross-reference format in a project

1. On the **Project** menu, select the target next to **Active Target** for which you want to modify cross-reference formats.
2. On the **Target** menu, click **Cross Reference Rules**. You must have target modification permissions to modify a cross-reference format for a target. For more information, see “Customizing Target Settings”.
3. Specify the appropriate value for each cross reference. For more information about cross reference values, click **Help**.
4. Click **OK**.
5. Generate your output. For more information, see “Generating Output”.
6. Review your output and confirm that cross-references display appropriately in your generated output. For more information, see “Viewing Output”.

Adding Cross-Reference Formats to Projects

ePublisher obtains the cross-reference formats and values in the Cross-Reference Rules window from your source documents. You can also add cross-reference formats in ePublisher.

For example, if you started to use a new cross-reference format in your source document and the Stationery designer has not yet added this new cross-reference format to the Stationery associated with your project, you can add the new cross-reference format to your project and specify the cross-reference format you want to use for your new cross-reference format. After you add a new cross reference format ePublisher recognizes the new cross reference formats and applies the cross-reference format you specify.

To add a cross-reference format to a project

1. On the **Project** menu, select the target next to **Active Target** for which you want to add a cross-reference format.
2. On the **Target menu**, click **Cross Reference Rules**. You must have target modification permissions to add a cross-reference format for a target. For more information, see “Customizing Target Settings”.
3. In the **Document type** field, select the content authoring tool for the cross-reference format you want to add.
4. Click the **Add New Cross Reference** icon.
5. In the **Name** field, type a name for the new cross-reference format you want to add to the project.
6. In the **Replacement** field, type a combination of text and code or building blocks that define how you want your new cross-reference to display. For more information about cross-reference building blocks or codes, see your content authoring tool Help.
7. Click **OK**.
8. Click **OK** again to close the window.
9. Generate your output. For more information, see “Generating Output”.
10. Review your output and confirm that cross-references display appropriately in your generated output. For more information, see “Viewing Output”.

Deleting Cross-Reference Formats from Projects

ePublisher obtains the cross-reference formats and values in the Cross-Reference Rules window from your source documents. You can delete cross-reference formats in ePublisher. Delete cross-reference formats when you no longer want to use the cross-reference format in your source documents.

If you delete the cross-reference format in your ePublisher project, but your source documents continue to use the cross-reference format, ePublisher will detect the deleted cross-reference format in your source documents and add it to your project again the next time you scan your source documents or generate output.

To delete a cross-reference format from a project

1. On the **Project** menu, select the target next to **Active Target** for which you want to delete a cross-reference format.
2. On the **Target** menu, click **Cross Reference Rules**. You must have target modification permissions to delete a cross-reference format for a target. For more information, see “Customizing Target Settings”.
3. In the **Document type** field, select the content authoring tool associated with the cross-reference format you want to delete.
4. In the **Name** column, select the cross-reference format you want to delete.
5. Click the **Delete Cross Reference** icon.
6. Click **OK**.

Customizing File Mappings

This section explains how to customize file mappings in a project.

Understanding File Mappings

In ePublisher, a file mapping is an association between a file extension and an ePublisher adapter. An ePublisher adapter is an ePublisher component that links the content authoring tool that you used to develop your content with ePublisher. ePublisher currently provides adapters for the following content authoring tools:

- Adobe FrameMaker
- Microsoft Word
- XML

In ePublisher, you can add any source documents that can be opened with Adobe FrameMaker, Microsoft Word, DITA-XML to your ePublisher project through the use of file mappings. By default, ePublisher provides a list of file extensions that are preset to use either Microsoft Word, Adobe FrameMaker, or the built-in XML adapter. For example, you can add `.txt` files to your ePublisher project by specifying the adapter ePublisher should use in order to open the `.txt` file. You can specify whether you want the Adobe FrameMaker, Microsoft Word, or XML adapter to open the `.txt` files you add to your project.

Certain file extensions, such as `.book`, `.fm`, and `.bk` files, are unique to a specific adapter. For example, `.book`, `.fm`, and `.bk` file can only be opened by Adobe FrameMaker. `.rtf`, `.xml`, and `.doc` are specific to Microsoft Word. If you try to generate output or an output preview using a file type associated with an ePublisher adapter and the file type cannot normally be opened with the content authoring tool associated with the ePublisher adapter, ePublisher displays an error message. The built-in XML adapter ePublisher provides is configured out-of-the-box to support DITA-XML. You can also configure ePublisher Stationery to support other XML types. However, XML input formats other than DITA-XML may not be supported by the WebWorks Technical Support team.

If you have an ePublisher Contract ID that enables only the Microsoft Word, the Adobe FrameMaker, or the built-in XML adapter, then you can use only that adapter when you use ePublisher. Although the option to choose another adapter may be available in the ePublisher user interface, you will not be able to generate output or preview output using the other adapters. You can only use the adapters enabled by your Contract ID.

Modifying File Mappings

ePublisher provides a default list of file mappings in which file extensions have been preset to use a specific adapter. However, in some cases you may need to modify file mappings for a project. For file extensions that can either be opened with Microsoft Word or Adobe FrameMaker, such as `.txt` files, you can specify the adapter you want ePublisher to use for the file extension. You can modify file mappings for a specific project or for all of your ePublisher projects.

To modify a file mapping

1. *If you want to modify a file mapping for a specific project*, complete the following steps:
 - a. On the **Project** menu, click **Project Settings**.
 - b. In the **File Extension** column, click the file extension for which you want to modify the file mapping.
2. *If you want to modify a file mapping for all of your ePublisher projects*, complete the following steps:
 - a. On the **Edit** menu, click **Preferences**.
 - b. On the **File Mappings** tab, in the **File Extension** column, click the file extension for which you want to modify the file mapping.
3. In the **Adapter** column, select the ePublisher adapter you want to associate with the file extension. The ePublisher adapter you associate with the file extension will be the ePublisher adapter that opens files with the specified file extension.
4. Click **OK**.
5. Click **OK** again. Each new ePublisher project you create after you modify the file mapping will use the ePublisher adapter you associated with the file extension.

Creating New File Mappings

If there is a file extension that you would like to use but the file extension is not available in ePublisher in the default list of file extensions, you can create a new file mapping. To create a new file mapping, add a new file extension and associate, or map, the file extension to an ePublisher adapter. You can use the new, or custom, file mapping to specify that ePublisher open files using the new file extension with Adobe FrameMaker, Microsoft Word, or the built-in XML adapter. When you create a new file mapping, ePublisher saves information about the new file mapping you created, and you can apply the new file mapping to all of the subsequent projects that you open.

When you create a file mapping and specify an ePublisher adapter for the file extension, ensure the file extension can be opened using the content authoring tool associated with the adapter outside of ePublisher before you create the new file mapping. If the file extension cannot be normally opened using the content authoring tool, then ePublisher will also not be able to generate output from the source document using the ePublisher adapter.

For example, assume your Contract ID enables licensing for ePublisher Express for FrameMaker. Next assume that you add HTML as a file mapping and associate the .html file extension with the Microsoft Word adapter. When you create the file mapping for the .html file extension with the Microsoft Word adapter, ePublisher allows you to add the HTML file to your project. However, since you do not have a valid license key for the ePublisher Express for Microsoft Word, ePublisher displays the following error message.



To create a new file mapping

1. *If you want to create a new file mapping for a specific project*, complete the following steps:
 - a. On the **Project** menu, click **Project Settings**.
 - b. Click the **Add** icon.
2. *If you want to create a new file mapping for all of your ePublisher projects*, complete the following steps:
 - a. On the **Edit** menu, click **Preferences**.
 - b. On the **File Mappings** tab, click the **Add** icon.
3. In the **File extension** field, type the file extension you want to use for the file mapping. For example, you can add .html as a file extension.
4. In the **Adapter** field, select the ePublisher adapter you want to associate with the file extension. The ePublisher adapter you associate with the file extension will be the ePublisher adapter that opens files with the specified file extension. For example, you can select Microsoft Word as the adapter for the .html file extension.
5. Click **OK**.
6. Click **OK** again. Each new ePublisher project you create after you modify the file mapping will use the ePublisher adapter you associated with the file extension.

Deleting File Mappings

Delete a file mapping when you no longer want to use the file mapping in your ePublisher project.

To delete a file mapping

1. *If you want to delete a file mapping for a specific project*, complete the following steps:
 - a. On the **Project** menu, click **Project Settings**.
 - b. In the **File extension** field, select the file extension for the file mapping you want to delete.
2. *If you want to delete a file mapping for all of your ePublisher projects*, complete the following steps:
 - a. On the **Edit** menu, click **Preferences**.
 - b. On the **File Mappings** tab, in the **File extension** field, select the file extension for the file mapping you want to delete.
3. Click the **Delete** icon.
4. Click **OK**.
5. Click **OK** again. Each new ePublisher project you create after you delete the file mapping will not use the ePublisher adapter you deleted.

11

Scheduling and Integrating Processes with AutoMap

This section describes how to automate output generation and integrate this task with your other processes. For example, you can have the build process for a software product automatically build all the targets you define in your project using the latest version of source documents checked into your version control system or content management system.

How ePublisher Supports Automation

ePublisher provides several components to help you process source documents and publish the generated output. The ePublisher AutoMap component helps you schedule and perform all your processing tasks, and it also provides automated pre- and post-processing capabilities to complete your production and publication processes.

What Is ePublisher AutoMap?

ePublisher AutoMap is the automation tool that enables you to automate the content transformation process, batch processing, and integration with content management or version control systems. This component lets you schedule ePublisher projects. For example, you can schedule the output generation to occur overnight. Then, when you arrive the next morning, your transformed content is ready for you. You can also automatically generate and deploy deliverables to meet your specific needs, such as updating Web site content based on updated source documents. You can automatically create ePublisher projects and generate output without manually opening ePublisher or your source documents.

Benefits of Using ePublisher AutoMap

ePublisher AutoMap automates the process of transforming your source documents to your output formats. This component lets you transform content at scheduled times and work seamlessly with your content management and version control systems. The following list highlights several ePublisher AutoMap features that save you time and effort:

- Automates the output generation using existing projects, including synchronizing with the Stationery
- Creates projects using the specified Stationery and applies it to your content
- Creates merged output from multiple books
- Allows you to customize conditions, variables, and cross references on a per-job and per-target basis
- Redirects and deploys output automatically
- Provides a full-featured command-line interface for performing batch transformations from other systems or scripts
- Integrates with content management and version control systems
- Offers flexible scheduling options
- Notifies relevant people when a job succeeds or fails
- Automatically updates your online content, help, or Web-based information

Version Control System (VCS) Integration

ePublisher AutoMap allows you to specify scripts for retrieving files from version control systems such as:

- CVS
- ClearCase
- Visual Source Safe
- Subversion
- Mercurial
- Documentum
- Perforce
- Git

ePublisher AutoMap can work with any version control system that is scriptable from the command line. For more information, see “CVS Version Control Checkout Example”. To see more version control integration scripting examples see wiki.webworks.com/HelpCenter/Tips/VersionControl

Content Management System (CMS) Integration

Using the same scripting integration as used for version control system integration, ePublisher AutoMap can also work with many industry standard content management systems such as: Vasont CMS and SDL LiveContent.

Preparing Projects, Stationery, and Source Files

ePublisher AutoMap can transform your source documents with no special modifications. Prepare your project, Stationery, and source documents just like you do when you generate output with the other ePublisher components. In ePublisher AutoMap, you use a job to define the output generation and the applicable options. After you create your job, ePublisher AutoMap performs one of the following tasks:

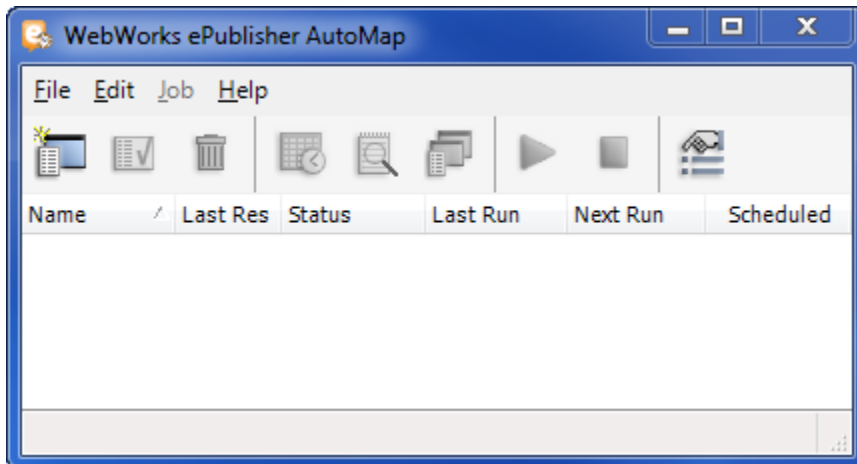
- If you created the job based on a project, ePublisher AutoMap synchronizes the project with the Stationery, and then it generates the output defined by the project.
- If you created the job based on a Stationery, ePublisher AutoMap creates a new project based on the Stationery, adds all the source documents defined by the job, and then it generates the output defined by the project.

Starting ePublisher AutoMap

ePublisher AutoMap features an easy-to-use console that allows you to schedule output generation using your source documents to run either immediately or at a specified time.

To start ePublisher AutoMap

In the **WebWorks** program group, click **ePublisher AutoMap > WebWorks ePublisher AutoMap**.



To start ePublisher AutoMap without administrative privileges

If are running in a restrictive environment that does not allow elevation of user permissions to Administrator rights, then you will need to run ePublisher AutoMap from the following batch file. This file lives in the installation directory underneath `WebWorks\ePublisher\<version>\ePublisher AutoMap\`. The file is called `WebWorks.Automap.Administrator.restricted.bat`.

Note: You will not be able to launch or schedule jobs when starting ePublisher AutoMap this way. However you will be able to create and configure jobs.

Setting ePublisher AutoMap Preferences

The Preferences window lets you specify default options and preferences that affect the behavior of ePublisher AutoMap. These preferences allow you to customize the console for your specific needs.

Specifying the Job, Staging, and User Formats Folder Locations

ePublisher AutoMap has several folders where it stores job-related information, such as job files and log files. You can customize the location of these folders.

Job Folder

ePublisher AutoMap stores the following types of job-related files in the job folder:

log file

A text file that contains information about the last transformation process. ePublisher AutoMap creates the log file when you run the job. You can view the log file with any text editor.

job file

A proprietary XML formatted file that contains information describing the job. When you create a job, ePublisher AutoMap creates the job file and stores the information you specify in the console. The job file name is the name of the job with a .waj (WebWorks AutoMap Job) file extension.

Note: ePublisher AutoMap maintains the job files. Do not edit these files.

The default job folder is the `My Documents\ePublisher AutoMap\Jobs` folder.

Staging Folder

ePublisher AutoMap stores information needed for transforming the source documents in the staging folder. This information includes the automatically generated ePublisher project, intermediate data files, and the output files. The staging folder provides a working folder for processing each job. ePublisher AutoMap then writes the final output files to the target output destination specified when the project was created.

Note: ePublisher AutoMap creates the files in the staging folder. Do not edit these files.

The default staging folder is the `My Documents\ePublisher AutoMap\Staging` folder.

User Formats Folder

When ePublisher AutoMap processes a job, it searches the user formats folder for custom user formats. The default user formats folder is the `My Documents\ePublisher Designer User Formats` folder.

To specify the job, staging, and user formats folder locations

1. Start ePublisher AutoMap.
2. On the **Edit** menu, click **Preferences**.
3. Click the **General** tab.
4. In the **Job Folder** field, type the path to the folder or network share where you want ePublisher AutoMap to store the job and log files. You can click **Browse** and navigate to the folder or network share.
5. In the **Staging Folder** field, type the path to the folder or network share where you want ePublisher AutoMap to store the job projects and output created during your transformations. You can click **Browse** and navigate to the folder or network share.
6. In the **User Formats Folder** field, type the path to the folder or network share where ePublisher AutoMap checks for the custom user formats to use for creating jobs. You can click **Browse** and navigate to the folder or network share.
7. Click **OK** to save your preferences.

Automatic Scanning for Conditions and Variables

When ePublisher AutoMap displays windows that list conditions and variables, you can specify whether ePublisher AutoMap automatically scans the source documents for updated conditions and variables. Since your source documents may not have new conditions or variables, ePublisher AutoMap attempts to save time by default and it does not automatically scan for new conditions and variables. You can click **Scan Documents** on a window to have ePublisher AutoMap scan your source documents and update the window contents.

If you want ePublisher AutoMap to always display updated condition and variable values, select the **Always scan for variables and conditions** option. Depending on the number and size of files included in the job, each scan may require several minutes before ePublisher AutoMap can display the conditions and variables windows.

To enable automatic scanning

1. Start ePublisher AutoMap.
2. On the **Edit** menu, click **Preferences**.
3. On the **General** tab, select **Always scan for variables and conditions**.
4. Click **OK** to save your preferences.

Keeping or Deleting Temporary Files

You can choose whether ePublisher AutoMap keeps or deletes the temporary files created in the staging folder when it runs a job. By default, ePublisher AutoMap does not delete these files. You need these files only if you want to examine the actual project and intermediate files created and used during the transformation process. If you do not want to examine these files, you can disable allow ePublisher AutoMap to delete the files and reduce the amount of disk space used.

To delete temporary files

1. Start ePublisher AutoMap.
2. On the **Edit** menu, click **Preferences**.
3. On the **General** tab, select **Delete temporary files after generating**.
4. Click **OK** to save your settings.

Defining File Mappings

ePublisher allows you to map files based on their file extension to a source document adapter that processes that input format. This capability provides the flexibility you need:

- You can define which source application processes a file that can be handled by multiple source applications. For example, both Adobe FrameMaker and Microsoft Word can open text files with a `.txt` extension. The File Mappings tab allows you to choose which application to use for each file based on the file extension.
- You can define custom file extensions. For example, you may choose to use a non-standard file extension for your Microsoft Word source documents, such as `.word` instead of `.doc`. The File Mappings tab allows you to add your custom file extension and map it to the appropriate source application adapter.
- You can add new input formats as they become available. ePublisher provides a powerful framework in which Quadralay and their partners can develop new input formats, such as custom XML formats. Since XML is a common format, the source files are usually given unique and descriptive file extensions. The File Mappings window ensures that your projects are ready to handle any input sources.

Note: ePublisher AutoMap lists only the source application adapters installed with the product. Your Contract ID specifies the adapters for which you are licensed. adapter you want to use.

To modify your file mappings

1. Start ePublisher AutoMap.
2. On the **Edit** menu, click **Preferences**.
3. Click the **File Mappings** tab.
4. *If you want to add a new file mapping*, complete the following steps:
 - a. Click **Add**.
 - b. In **File extension**, type a file extension to be mapped.
 - c. In **Adapter**, select one of the installed source adapters, and then click **OK**.
5. *If you want to modify an existing file mapping*, in the **Adapter** column next to the file extension you want to modify, select the installed source adapter you want to use for files with that file extension.
6. *If you want to delete an existing file mapping*, complete the following steps:
 - a. In the **File Extension** column, select the file extension you want to delete.
 - b. Click **Delete**.
7. Click **OK** to save your preferences.

Defining Output Destinations

ePublisher AutoMap allows you to create and manage output locations independent of your jobs. This flexibility allows you to define your output locations and then select the one you want to use when you create a new job. You can also create a new output location as the final part of the job creation process. ePublisher AutoMap allows you to deploy the final output files to a folder on a local or shared file system, such as `C:\helpdocs` or `\\server\share`.

To define output destinations:

1. *If you use the ePublisher AutoMap user interface to schedule an ePublisher Express project*, you must set the deployment destination in the ePublisher Express project. For more information, see “Creating Output Destinations”.
2. *If you use the ePublisher AutoMap CLI to run an ePublisher Express project*, you can set the deployment destination in the ePublisher Express project or in the CLI. For more information, see “Creating Output Destinations” and “Using the Command-Line Interface”.
3. *If you use ePublisher AutoMap to schedule a Stationery project*, you must set the deployment destination in the job settings, and you must set the deployment destination independently for each target. Specify the deployment destination in the job settings by completing the following steps:
 - a. Start ePublisher AutoMap.
 - b. Select the job to edit in the ePublisher AutoMap main window.
 - c. On the **Job** menu, click **Edit**.
 - d. Select the Target Configuration tab.
 - e. In the left pane, select the target for which you want to set the output destination.
 - f. On the Info tab, in the **Deployment** area, specify the deployment destination information.
 - g. Click **OK**.
4. *If you use the CLI to run a Stationery project*, you can set the deployment destination in the ePublisher AutoMap job settings or in the CLI. Set the deployment destination in the job settings by completing the following steps:
 - a. Start ePublisher AutoMap.
 - b. Select the job to edit in the ePublisher AutoMap main window.
 - c. On the **Job** menu, click **Edit**.
 - d. Select the Target Configuration tab.
 - e. In the left pane, select the target for which you want to set the deployment destination.
 - f. On the Info tab, in the **Deployment** area, specify the deployment destination information.
 - g. Click **OK**.

For more information about setting the deployment destination in the CLI, see “Using the Command-Line Interface”.

Defining Email Notifications

ePublisher AutoMap can send an email notification after running a job. The notification provides information about whether the job ran successfully or encountered any errors. You can also specify whether to include the log file as a text file attachment to the email.

To configure email notification

1. Start ePublisher AutoMap.
2. On the **Edit** menu, click **Preferences**.
3. Click the **Notification** tab.
4. Select **Enable Email Notification**.
5. Specify the email addresses and email server information. For more information about a field, click **Help**.
6. Click **OK** to save your preferences.

Note: You can quickly enable or disable email notification without modifying the other preferences by toggling the **Enable email notification** check box.

Selecting Console Language (English, German, French, and Japanese)

In addition to the English language, the ePublisher consoles have been translated into German, French, and Japanese languages. When ePublisher starts, it detects the operating system locale and displays the corresponding ePublisher console. You can display the console for a locale that is not the operating system default. For example, you can display the German console on an English version of Windows.

To select the language for the console

1. Start your ePublisher console.
2. On the **Edit** menu, click **Preferences**.
3. In the **User interface language** field on the **General** tab, select the language you want the console to display.
4. Click **OK** to save your preferences.
5. Close and restart the console to view the console in the selected language.

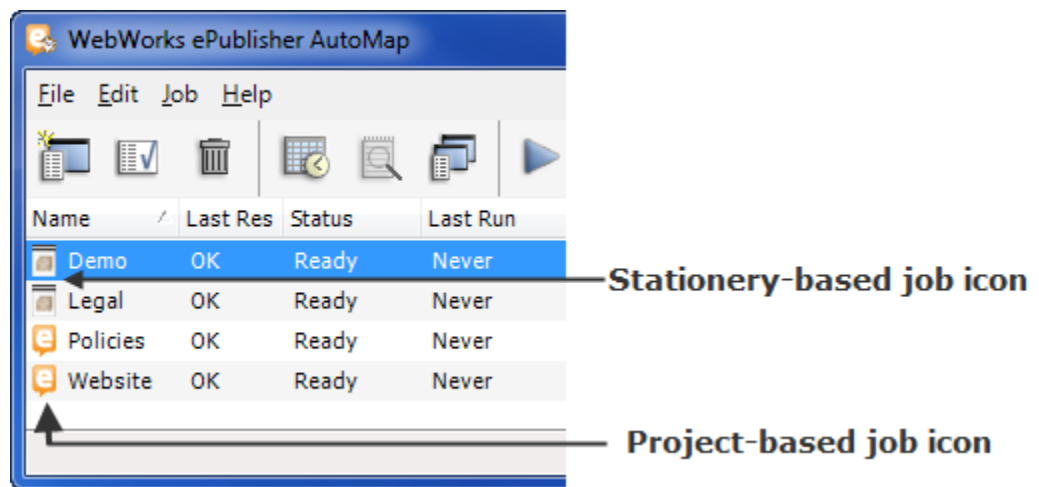
Working with Jobs

This section describes how to create and manage ePublisher AutoMap jobs. A job is a set of tasks that ePublisher AutoMap can perform based on a Stationery or a project. You can immediately run a job, or you can schedule a job to run at a later time. By default, job files are stored in the following folder:

```
My Documents\WebWorks ePublisher AutoMap\Jobs
```

You can change the default location in the ePublisher AutoMap preferences. Job files have a .waj file extension. Job files depend on other files located in the job folder and should not be moved or copied independently.

ePublisher AutoMap displays a unique icon to help you identify project-based jobs and Stationery-based jobs. Each job name is preceded by the corresponding icon.



After you schedule a job in the ePublisher AutoMap user interface, you can close the ePublisher AutoMap user interface, and the job will still run based on the schedule you specified.

When ePublisher AutoMap runs a job either from a schedule or from a command prompt, Windows opens a command prompt that displays the log as it is generated. When the job finishes, the command prompt closes. After the job runs, you can view the log file using one of the following methods:

- **If you use ePublisher AutoMap to schedule an Express or Stationery project**, the log is stored in the Job folder. You can view the log by clicking **View log** on the **Job** menu in the ePublisher AutoMap user interface.
- **If you use the CLI to run an Express project**, there is no entry in the ePublisher AutoMap user interface. You can find the log in the Express project directory and view it in Notepad.
- **If you use the CLI to run a stationery project**, the log is stored in the Job folder. You can view the log by clicking **View log** on the **Job** menu in the ePublisher AutoMap user interface.

Note: The ePublisher AutoMap user interface displays the last time that the Windows Scheduler ran the stationery project. It does not display the time the CLI ran.

Creating a Project-Based Job

Project-based jobs allow you to schedule ePublisher AutoMap to generate the output defined by an existing ePublisher project. You first create the ePublisher project as needed, or you can use an existing project. Then, you create a project-based job so ePublisher AutoMap can use the project to generate the output defined by the project.

To create a project-based job

1. Start ePublisher AutoMap.
2. On the **File** menu, click **New Job**.
3. Click **ePublisher project** on the New Job window.
4. Type the path or click the **Browse Folder** icon to select the ePublisher project you want to schedule, and then click **OK**.
5. In the **Job Name** field, type the desired job name.

Note: The **Choose ePublisher project** field displays the previously selected ePublisher project. You can change the selected project.

6. *If you want to run a pre- or post-build script before or after the job runs*, complete the following steps:
 - a. Click the **Edit Script** button for the **Pre-build** or **Post-build** script field.
 - b. Type or paste your script into the editor, and then click **OK**. For more information about scripts, see “Using Scripts for Additional Custom Processing”.
7. Click **Next**.

ePublisher displays the Target Selection window. This window lets you select which targets to generate as part of this ePublisher AutoMap job. Depending on your project, there may be multiple targets listed in the Target Selection window.

8. Select the check box in the **Build** column next to each target you want to generate, and then click **Finish**.
9. Schedule the job as needed, and then click **OK**. You can also schedule the job at a later time. For more information, see “Scheduling Jobs with Windows Scheduler”.

Project-based jobs deploy output only if a deployment target is set in the project itself. This deployment information is specified in the target settings in ePublisher. ePublisher AutoMap reads only the deployment target name from the project. Therefore, you must make sure an output destination with the same name and is defined in ePublisher AutoMap or you may receive a deployment error. For more information, see “Defining Output Destinations”.

Creating a Stationery-Based Job

Stationery-based jobs provide a powerful solution that can save your time and increase productivity. These jobs can also reduce errors when generating online and print content. Stationery-based jobs allow you to associate your source documents with your ePublisher Stationery. This combination lets you apply a single Stationery to many different source documents, reusing the transformation defined by the Stationery. This method ensures accuracy and consistency across many different projects and output files for your organization.

To create a Stationery-based job

1. Start ePublisher AutoMap.
2. On the **File** menu, click **New Job**.
3. Select **ePublisher Stationery** on the New Job window.
4. Type the path or click the **Browse Folder** icon to select the Stationery upon which you want to base this job, and then click **OK**.
5. In the **Job Name** field, type the desired job name.

Note: The **Choose ePublisher Stationery** field displays the previously selected Stationery. You can change the selected Stationery.

6. *If you want to run a pre- or post-build script before or after the job runs*, complete the following steps:
 - a. Click the **Edit Script** button for the **Pre-build** or **Post-build** script field.
 - b. Type or paste your script into the editor, and then click **OK**. For more information about scripts, see “Using Scripts for Additional Custom Processing”.
7. Click **Next**.

ePublisher displays the Documents window. This window lets you configure the groups and documents you want to generate output for in this job. You can either manually add the documents or you can run a script to retrieve documents within a group.

For example, you may want to retrieve documents from a version control or content management system. You can provide a script to retrieve and prepare your documents as needed. This script runs before the transformation, which ensures that you always have the most current version of your content without having to perform a manual update before each transformation.

8. *If you want to manually add groups of documents to the job*, complete the following steps:
 - a. Click **New Group**.
 - b. Specify a name for the group.
 - c. Click **Add Document**, and then browse and select your source documents.
 - d. Repeat this process to add more groups and source documents, and then click **Next**.
9. *If you want to use a script to add a group of documents*, complete the following steps:
 - a. Click **New Group**.
 - b. Specify a name for the group.
 - c. Click **Edit Script**.

- d. Type or paste your script into the editor, and then click **OK**.

10. Click *Next*.

ePublisher displays the Target Selection window. This window lets you select which targets to generate output for as part of this ePublisher AutoMap job.

11. Select the check box in the *Build* column next to each target you want to generate, and then click *Next*.

ePublisher displays the Target Configuration window. The Stationery defines the configuration options for each target. You can override these configuration options for each target in a Stationery-based job. Depending on the output format of the selected target, the Target Configuration window provides tabs for adjusting various options, such as conditions and variables.

The information presented on each tab is relative to the target selected in the **Target Name** column. You can adjust values for each target independent of the other targets. For example, you can have two WebWorks Help 5.0 targets where the conditions are set differently depending on the target audience.

12. *If you want to override the configuration settings defined in the Stationery for a target*, complete the following steps:

- a. Select the target you want to modify in the **Target Name** column.
- b. Specify the appropriate values for the configuration options on each tab for that target. For more information about the fields on a tab, click **Help**.

13. Click *Finish*.

14. Schedule the job as needed, and then click *OK*. You can also schedule the job at a later time. For more information, see “Scheduling Jobs with Windows Scheduler”.

Duplicating an Existing Job

Sometimes it is more convenient to make a copy of an existing job and adjust its options instead of creating a new job. ePublisher AutoMap allows you to duplicate an existing job. Then, you can modify the options as needed for your new job.

To duplicate an existing job

1. Start ePublisher AutoMap.
2. Select the job you want to copy in the ePublisher AutoMap main window.
3. On the **Job** menu, click **Duplicate**.
4. Specify the new job name, and then click **OK**.
5. Specify the scheduling options as needed, and then click **OK**. For more information, see “Scheduling Jobs with Windows Scheduler”.

Once the new job exists, you can modify the job for your specific needs. For more information, see “Editing an Existing Job”.

Editing an Existing Job

When you edit an existing job, ePublisher AutoMap presents the windows used to create the job as tabs of the Edit Job window. Select the appropriate tab that contains the information you want to modify.

To edit an existing job

1. Start ePublisher AutoMap.
2. Select the job to edit in the ePublisher AutoMap main window.
3. On the **Job** menu, click **Edit**.
4. Select the appropriate tab and modify the values as needed. For more information about a field, click **Help**.
5. Click **OK**.

Scheduling Jobs with Windows Scheduler

Immediately after creating a new job, ePublisher AutoMap starts the Windows Scheduler to allow you to schedule your job. ePublisher AutoMap uses the Windows Scheduler built into the Microsoft Windows operating system. The Windows Scheduler allows you to schedule a job to run at pre-determined times and repeating intervals. If you do not want to schedule the job, click **Cancel**. You can schedule the job at a later time, and you can modify an existing schedule.

When you schedule a job and then click **OK**, Windows prompts you for your Windows user name and password. For more information about using the Windows Scheduler, see the Windows operating system online help. To open the Windows online help, click **Help and Support** on the Start menu.

To schedule a job or change the schedule for an existing job

1. Start ePublisher AutoMap.
2. Select the job to schedule or reschedule in the ePublisher AutoMap main window.
3. On the **Job** menu, click **Schedule Job**.
4. *If you want to create a schedule for a job with an existing schedule*, click **Show multiple schedules**, and then click **New**.
5. *If you want to create a new schedule for a job that is not currently scheduled*, click **New**.
6. Specify the appropriate values, such as the **Start time**, and then click **OK**.
7. Specify your Windows user name and password, and then click **OK**. Since you are scheduling a task in Windows Scheduler, you must provide your Windows user name and password to add the task. If you are part of a Windows domain, include the domain name, such as *domain\user*.

Deleting an Existing Schedule for a Job

You can create multiple schedules for a job. You can also delete one or more of these existing schedules.

To delete an existing schedule for a job

1. Start ePublisher AutoMap.
2. Select the job to delete the schedule for in the ePublisher AutoMap main window.
3. On the **Job** menu, click **Schedule Job**.
4. Select the **Show multiple schedules** check box.
5. Select the schedule you want to delete, and then click **Delete**.
6. Click **OK** to close the Scheduler window.
7. Specify your Windows user name and password, and then click **OK**. Since you are changing a task schedule in Windows Scheduler, you must provide your Windows user name and password. If you are part of a Windows domain, include the domain name, such as *domain\user*.

Running an Existing Job

You can run a job at any time whether it is scheduled or not. This feature allows you to test the job while you are configuring it. You can also run a job to quickly generate output based on a set of predefined settings.

To run an existing job

1. Start ePublisher AutoMap.
2. Select the job you want to run in the ePublisher AutoMap main window.
3. On the **Job** menu, click **Run**.

Viewing a Job Log File

When a job runs, ePublisher AutoMap creates a log file that documents the steps performed for that job. You can view this log file when one exists. ePublisher AutoMap keeps the log file only for the last time the job ran. Previous log files are not kept.

To view a log file

1. Start ePublisher AutoMap.
2. Select the job to view the log for in the ePublisher AutoMap main window.
3. On the **Job** menu, click **View Log**.

Canceling a Job

When you cancel a running job, ePublisher AutoMap, Microsoft Windows, Adobe FrameMaker, and Microsoft Word must do some clean-up work. During this process, Windows may display the standard Microsoft Windows End Program window. Do *not* click the **End Now** or **Cancel** buttons. The window will close usually in less than 15 seconds once everything has finished properly.

To cancel a running job

1. Start ePublisher AutoMap.
2. Select the running job you want to cancel in the ePublisher AutoMap main window.
3. On the **Job** menu, click **Stop**.

Deleting an Existing Job

If you no longer need a job, you can delete it from the ePublisher AutoMap main window.

To delete an existing job

1. Start ePublisher AutoMap.
2. Select the job to delete in the ePublisher AutoMap main window.
3. On the **Job** menu, click **Delete**.
4. Click **Yes** to confirm that you want to delete the selected job.

Using Scripts for Additional Custom Processing

You do not need to write scripts to use ePublisher AutoMap. However, scripts allow you to extend ePublisher AutoMap and integrate it with other products and custom workflows. You can define scripts to run before and after a job generates content, to run before and after each target generates content, and to retrieve source documents on a per group basis.

Writing Scripts

You can write scripts directly in the ePublisher AutoMap script editor window, or you can cut and paste scripts from another text editor. The ePublisher AutoMap script editor supports only text-based scripts. Any formatting or additional information available in a third-party script editor is lost when the script is pasted into the ePublisher AutoMap script editor.

ePublisher AutoMap treats scripts like DOS batch files. The script must be complete and valid or it will fail when the job runs. Although you can write all your scripts directly in the ePublisher AutoMap script editor; this approach may not be your best option.

When calling complex scripts, you may want to create and store those complex scripts in your file system and simply call those files from the script you create in the ePublisher AutoMap script editor. Since the script editor treats the scripts like DOS batch files, you can call other batch files, applications, or scripts written in any scripting language, such as VBScript, Perl, and Python. You can also pass parameters and script variables to the files you call. However, some variables have meaning only for certain types of scripts. For example, the DeployFolder variable is specific to a target and does not have meaning in a pre-build or post-build job script.

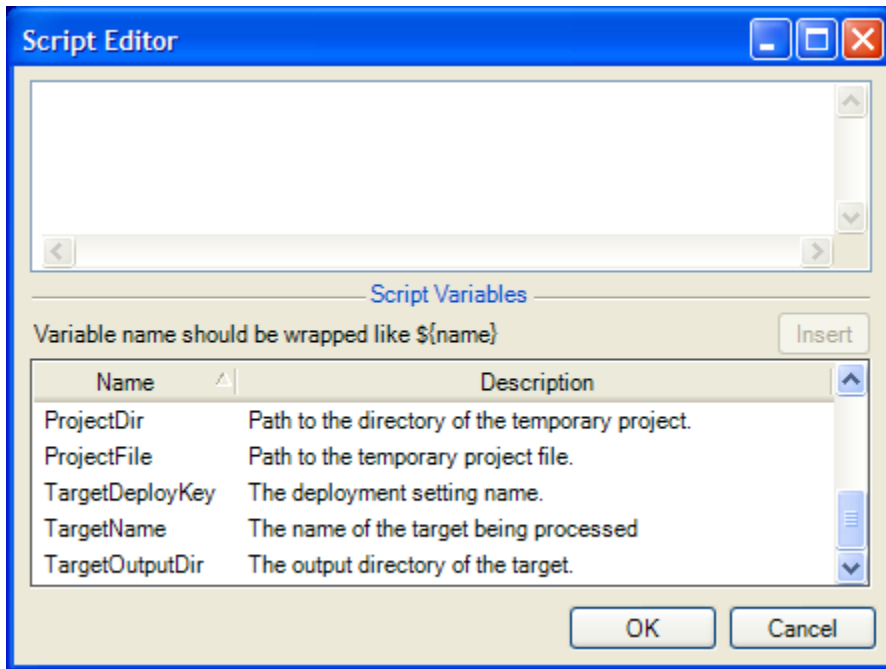
Working Folder

Windows associates batch files with a working folder. The working folder is the current working directory used by an application or batch file for processing. For example, if a batch file creates a new file without specifying a path, the file is created in the working folder. If a batch file attempts to read a file without specifying a path, Windows assumes the file is in the working folder.

The working folder for an ePublisher AutoMap batch file is the job folder itself. You can locate this folder using the job directory `${JobDir}` scripting variable. For more information, see “Using Scripting Variables Example”.

Opening and Using the Script Editor

ePublisher AutoMap includes an editor for specifying and writing job scripts. This editor provides a text area for writing or pasting your script and a list of ePublisher AutoMap variables that you can use in your scripts or pass to other scripts and applications.



To open the script editor and edit a script

1. Start ePublisher AutoMap.
2. Select the job to edit in the ePublisher AutoMap main window.
3. On the **Job** menu, click **Edit**.
4. Click the appropriate **Edit Script** button to open the script editor window for the script you want to edit.
5. Click in the text editing area.
6. Type your script, or press Ctrl+V to paste it from the clipboard.
7. *If you want to insert an ePublisher AutoMap scripting variable*, complete the following steps:
 - a. Click in the editor window where you want to insert the variable.
 - b. Double-click the variable name in the **Script Variables** pane to insert it. You can also type the variable name surrounded by `${}`. For example, to include the project directory variable in your script, type `${ProjectDir}`.

Scripting Variables

The following table provides a summary of the available scripting variables.

Variable	Description	Scope
FileListName	Returns the name of the file that contains the list of source documents.	Document scripts only
FileListPath	Returns the path of the file that contains the list of source documents.	Document scripts only
GroupName	Returns the name of the documents group that is currently being processed.	Document scripts only
BuildAction	Returns whether the current build action is pre-build or post-build.	Job scripts, target scripts, and document scripts
JobDir	Returns the path of the job .waj file.	Job scripts, target scripts, and document scripts
JobFile	Returns the name of the job .waj file.	Job scripts, target scripts, and document scripts
JobName	Returns the name of the job.	Job scripts, target scripts, and document scripts
ProjectDir	Returns the path of the temporary project file that ePublisher AutoMap created for the job.	Job scripts, target scripts, and document scripts
ProjectFile	Returns the name of the temporary project file that ePublisher AutoMap created for the job.	Job scripts, target scripts, and document scripts
DeployFolder	Returns the deployment directory path.	Target scripts only
ErrorCount	Returns the number of errors reported during the generation process.	Target scripts only
TargetDeployKey	Returns the deployment target name.	Target scripts only
TargetName	Returns the name of the target being generated.	Target scripts only
TargetOutputDir	Returns the output path of the target.	Target scripts only

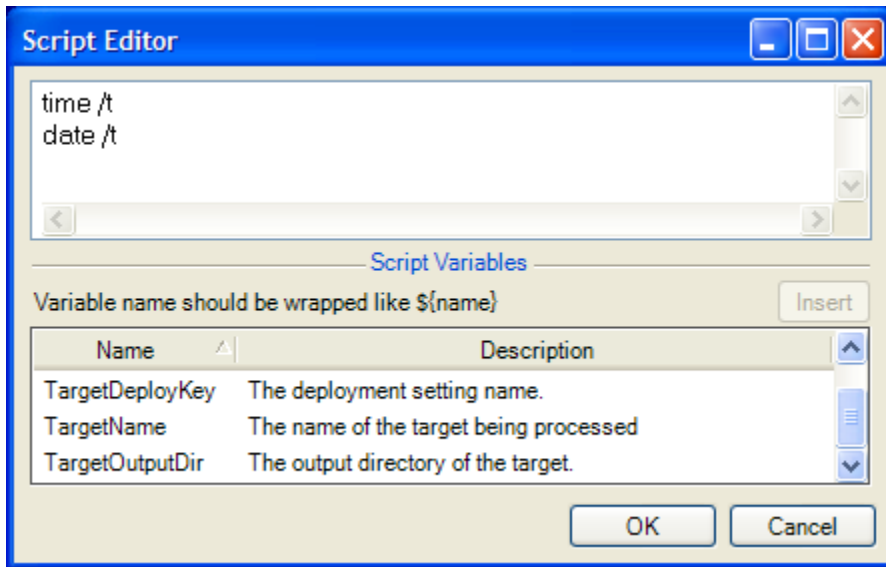
Scripting Examples

The following list highlights a few of the ways you can use scripts to customize and integrate the content publication process:

- You can write scripts to get the latest version of files from your version control system or content management system. For more information, see “CVS Version Control Checkout Example”.
- You can use variables within scripts. For more information, see “Scripting Variables” and “Using Scripting Variables Example”.
- You can customize the log file created when ePublisher runs a job. For more information, see “Show Time and Date Example” and “Using Scripting Variables Example”.
- You can use post-processing scripts to customize the deployment process. These scripts can modify files and even rebuild the final deliverable. For example, you can use scripts to further customize the generated files and then automatically run Microsoft HTML Help Workshop to rebuild the .chm file.

Show Time and Date Example

This example simply displays the time and date in the log file by calling the time and date commands built into the DOS command-line interface. In this case, the script was added as a pre-build step for the job running before the first target starts generating output.



The following output shows the time and date included in the log before the first target starts the generation process. This information can automatically timestamp a job log file. You can also use these values in your script to log elapsed time and make decisions.

```
WebWorks Automap
```

```
09:57 AM
```

```
Mon 6/23/2008
```

```
Scanning 1 document(s)...
```

```
Building target: WebWorks Help 5.0
```

```
Generation started at 9:57:15 AM
```

```
Initializing file information
```

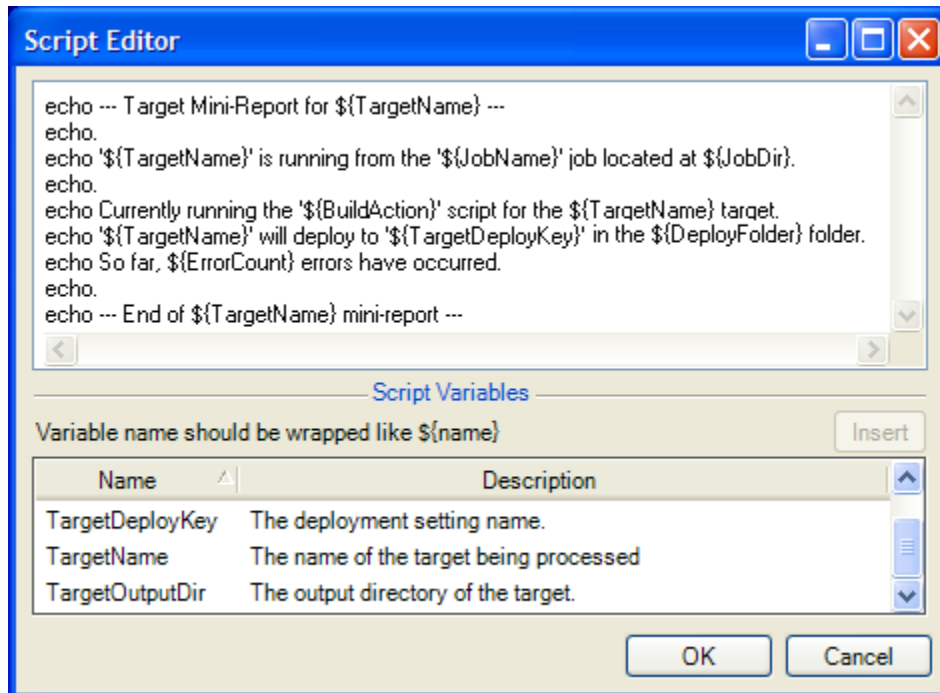
```
Updating documents.
```

```
Applying settings to WebWorks.doc, 1 of 1.
```

Using Scripting Variables Example

This example creates a simple mini-report inside the log file for each target in the job. This simple example demonstrates the use of the scripting variables. In this case, the values of the variables are displayed as part of the report using the echo command built into the DOS command-line interface.

The following figure shows some of the variable names surrounded by single quotes. You do not need to enclose variable names in quotes. These quotes are there only to show emphasis on the variable values in the mini-report example.



The following output shows the mini-report displayed immediately after the target starts to be built. You could add this same script as a post-build step for the target and a similar mini-report would be created when the target is done being generated.

```
Building target: WebWorks Help 5.0
```

```
--- Target Mini-Report for WebWorks Help 5.0 ---
```

```
'WebWorks Help 5.0' is running from the 'Scripting Demo' job located at C:\Documents and Settings\doc\My Documents\WebWorks Automap\Jobs\Demo.
```

```
Currently running the 'PreBuild' script for the WebWorks Help 5.0 target. 'WebWorks Help 5.0' will deploy to 'Online Help' located in the C:\AutoMapOutput\Help Systems\Online Help folder.
```

```
So far, 0 errors have occurred.
```

```
--- End of WebWorks Help 5.0 mini-report ---
```

```
Generation started at 9:57:15 AM
```

Initializing file information

Updating documents.

Applying settings to WebWorks.doc, 1 of 1.

CVS Version Control Checkout Example

The previous example scripts demonstrate some simple capabilities of using scripts in ePublisher AutoMap. This example shows how to check out source files from a version control system at the start of a job.

Version control systems let you store files and retain information concerning different versions of those files. You can store multiple versions of files, revert to previous versions of files, and share files among large groups of people. The advantage of using a version control script to retrieve source files rather than adding them directly to a project ensures that your job is always working with the latest checked in version of each source file.

Each version control system works in a slightly different way. ePublisher AutoMap is not tied to a specific version control solution. ePublisher AutoMap provides an abstract way of retrieving a list of source files to transform. There are a few rules about how this list is named and formatted, but the actual creation of the list is left to the script and scriptwriter.

The scripting capabilities in the Documents panel are not limited to checking files out of a version control system. The only requirement is that the `FileList.txt` file includes a list of files and paths to be transformed. You can access the `FileList.txt` file and the path to it using the ePublisher AutoMap `${FileListName}` and `${FileListPath}` scripting variables. For more information, see “Scripting Variables”.

The following script example calls `getfilesaction_cvs.vbs`, which is installed with ePublisher AutoMap. By default, this script is located in the following location:

```
\Program Files\WebWorks\ePublisher AutoMap\Scripts\getfilesaction_cvs.vbs
```

The `getfilesaction_cvs.vbs` script is written using the Visual Basic Scripting (VBS) capabilities built into Microsoft Windows to check out files from a CVS (Concurrent Version System) server and create a file list to be processed by ePublisher AutoMap. This script is not meant to be used in its current form. This script provides a starting point that includes everything you need to set up a working script for your particular environment.

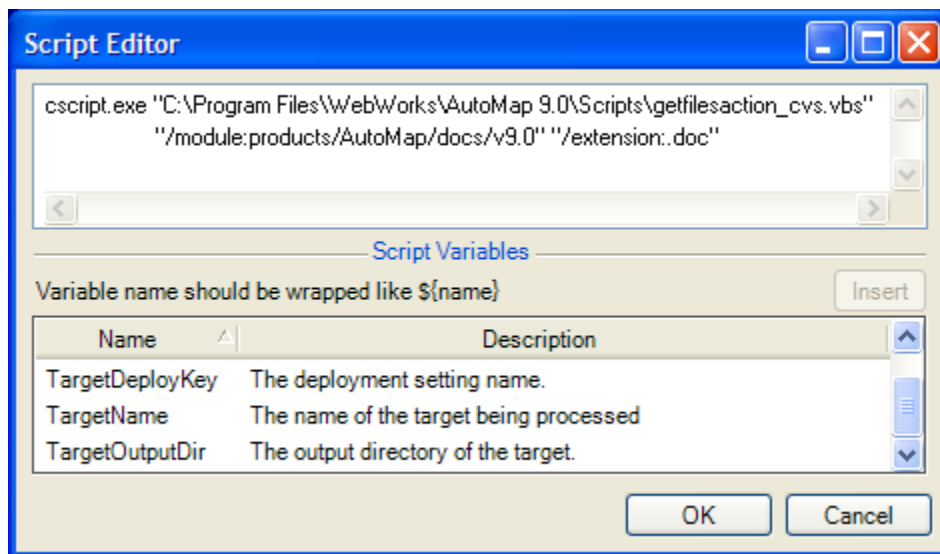
The `getfilesaction_cvs.vbs` script includes comments to help someone with scripting experience understand the actions it performs. Since this script provides only a starting point, you need to modify this script for your environment. For example, you need to set the CVS attributes for your user name, server name, and type of authentication. You also need to adjust for how CVS is installed and called in your environment. These areas of the script are highlighted in the following figure:

```

*   name and the CVSROOT parameters.
CVSRoot = ":pserver:username@machine:/remote_cvs_directory"
*
*   Main Script:
*   Checks out document files from CVS and creates the
*   file FileList.txt that contains each file that matches a specified
*   file extension.
*
*   This variable will hold a list of filenames and paths to be written to
*   FileList.txt. This list will be filled out in the GetMatchingFiles
*   function defined below.
Set VarFileList = CreateObject("Scripting.Dictionary")
*
*   Set the CVSROOT environment variable, which is required by cvs
*   so that it can locate files to be checked out.
GlobalEnv("CVSROOT") = CVSRoot
*
*   Set the current working directory so that the checked out files
*   will be placed into the correct directory.
GlobalShell.CurrentDirectory = CVSLocalDir
*
*   Execute the CVS command for checking out a module. Double-quotes are placed
*   around the CVSModule value to allow for spaces in the CVS module name passed in
*   on the command line.
*
*   NOTE: If the files being checked out are on a cvs branch, you must customize the
*   line below by removing the -A part of the command and replacing it with
*   -r <BranchName>.
*
*   NOTE: Also, if the path to cvs.exe is not listed in your system environment
*   variables, you will need to add the path to cvs to your system environment
*   variables or change the line below to point to the full path of cvs.exe
*   on your system. To test whether the call below will work, open a command
*   prompt window and type cvs at the prompt. If cvs is recognized by your
*   system, then the call below will work as is.
set ExecCall = GlobalShell.Exec("%comspec% /c cvs checkout -A " & chr(34) & CVSModule & chr(34))
*
*   Loop and sleep until the CVS checkout command is finished executing. The loop will
*   end if the execution completes or the loop counter reaches 11. If it is necessary
*   to wait longer for large CVS checkouts, you can customize the loop counter limit or
*   the sleep time between checks.

```

The following figure shows an example of how to call the `getfilesaction_cvs.vbs` script. The parameters passed indicate the module to checkout and to filter on the `.doc` extension. When this script runs, it checks out the indicated module from CVS, filters those files based on the `.doc` extension, and adds those `.doc` files to the `FileList.txt` file. That file list is then used by the job as the list of files to process and transform.



This example is only one possible way of creating a list of files to transform. Once you understand the mechanism that ePublisher AutoMap uses to retrieve its list of source documents to transform, you can create that list using the process you prefer. Whether you want to use a script or an application, ePublisher AutoMap can run it from the script editor as long as it is available to ePublisher AutoMap.

Using the Command-Line Interface

In addition to the ePublisher AutoMap console that allows you to create, edit, and schedule jobs, ePublisher AutoMap also provides a command-line interface (CLI). The CLI gives you control of ePublisher AutoMap from the Windows command-line interface. You can use the CLI and the supported options in scripts and batch files to automate and streamline your processes.

Running ePublisher AutoMap from the Command Line

You can run ePublisher AutoMap from the command line using the ePublisher AutoMap `WebWorks.AutoMap.exe` command-line application. This application is installed at the root of your ePublisher AutoMap installation location. The default location for this file is:

```
Program Files\WebWorks\ePublisher AutoMap\WebWorks.AutoMap.exe
```

Notes:

- If you install ePublisher AutoMap in another location, the `WebWorks.AutoMap.exe` file is installed in that location and you need to adjust your specified path.
- The `WebWorks.AutoMap.exe` file depends on other files installed in the ePublisher AutoMap folder. Do not move or copy the `WebWorks.AutoMap.exe` file.

To run ePublisher AutoMap from the command line

1. Open a Windows command prompt by completing the following steps:
 - a. On the **Start** menu, click **Run**.
 - b. Type `cmd`, and then click **OK**.
2. Enter the following command to change to the default ePublisher AutoMap folder:

```
CD \Program Files\WebWorks\ePublisher AutoMap
```
3. Enter the `WebWorks.AutoMap.exe` command with the appropriate options.

If you run the `WebWorks.AutoMap.exe` command without any options, ePublisher AutoMap displays a typical usage syntax statement.

CLI Syntax and Reference

The ePublisher AutoMap command-line interface can run a job or an ePublisher project. The command-line interface uses the following syntax:

```
WebWorks.Automap.exe [-f] [-n] [-c] [-l] [-u] [-j] [-d directory]  
[[-s directory] jobfile][[-t targets] projectfile]
```

The supported command-line options are defined in the following table.

Option	Description
-c	Deletes cached information and builds the output from scratch. Use this option only when passing an ePublisher project <i>.wrp</i> file. This option is equivalent to selecting Regenerate All . The default is to generate only what is needed and to use all available cached information. You can also specify this option as <code>--clean</code> .
-d <i>directory</i>	Specifies an alternate deployment directory. This setting overrides the specified default values. You can also specify this option as <code>--deployfolder <i>directory</i></code> .
-u	Scans all source documents for new styles, conditions, variables and xrefs. After the conversion is completed the resulting Express project file will be saved with all the updated information. You can also specify this option as <code>--update</code> .
-j	Just scans all source documents for new styles, conditions, variables and xrefs. Saves Express project file without running a conversion. You can also specify this option as <code>--justupdate</code> .
-f	Does not send an email notification when this job is finished. This setting overrides the default value specified in the ePublisher AutoMap preferences. You can also specify this option as <code>--nonotify</code> .
-l	Deletes all files in the deployment location before deploying the newly-generated output. This option ensures that all files in the deployment folder are from this last generation process. You can also specify this option as <code>--cleandeploy</code> .
-n	Generates output and does not deploy it. This setting overrides the default deployment settings specified in the job or ePublisher project. You can also specify this option as <code>--nodeploy</code> .
-s <i>directory</i>	Specifies the staging directory to use. Use this option only when passing a job <i>.waj</i> file. This setting overrides the default value specified in the ePublisher AutoMap preferences. You can also specify this option as <code>--stagingdir <i>directory</i></code> .
-t <i>targets</i>	Builds only the specified targets. Separate multiple targets with a comma and no space on either side of the comma. Use this option only when passing an ePublisher project <i>.wrp</i> file. The default is to build all targets. You can also specify this option as <code>--target <i>targets</i></code> .
<i>jobfile</i>	Specifies the name of the ePublisher AutoMap <i>.waj</i> job file.
<i>project</i>	Specifies the name of the ePublisher <i>.wrp</i> project file.

As with other command-line applications, specify the command-line options on the command line after the application name itself. Separate multiple command-line options with a space. Review the following additional notes concerning ePublisher AutoMap command-line options:

- The command-line options have both shortened and verbose formats. For example, you can specify the clean option as `-c` or `--clean`. Both options mean exactly the same thing and you can mix the shortened and verbose formats across the options.
- Some options require additional information. For example, the staging folder option requires the name of the directory and is specified as `--stagingdir directory`. You need to include the equals sign without spaces on either side.
- In cases where you specify paths with spaces, enclose those paths in double quotes to correctly handle the paths. You do not have to enclose paths without spaces in double quotes, but you can without an issue. For example, `c:\projects` works without enclosing it in double-quotes. However, `c:\my projects` does not work as expected. The correct format for the latter path is `"c:\my projects"`.

CLI Examples

The following examples illustrate how to use the command-line interface.

Running a Project and Updating the Express project file

The following example runs the `c:\Projects\MyProject.wrp` project and updates `MyProject.wrp` with the new styles, conditions, variables and xrefs:

```
WebWorks.AutoMap.exe -u "c:\Projects\MyProject.wrp"
```

Running a Project and Generating Only One Target

The following example runs the `c:\Projects\MyProject.wrp` project and generates only the `WebWorks Help` target:

```
WebWorks.AutoMap.exe -t "WebWorks Help" "c:\Projects\MyProject.wrp"
```

Running a Project from Scratch and Deploying to a Clean Location

The following example runs the `c:\Projects\MyProject.wrp` project and generates all targets defined in the project. This example also deletes the cached information in ePublisher to ensure it generates all the content from scratch, and it deletes all the files in the deployment location before deploying the newly-generated files:

```
WebWorks.AutoMap.exe --clean --cleandeploy "c:\Projects\MyProject.wrp"
```

Running a Project and Deploying to an Alternate Location

The following example runs the `c:\Projects\MyProject.wrp` project and generates all targets defined in the project. This example also deploys all the newly generated files to the `\\TestServer\Review` folder:

```
WebWorks.AutoMap.exe -d "\\TestServer\Review" "c:\Projects\MyProject.wrp"
```


Running a Job Without Sending Notification When Done

The following example runs the `c:\Jobs\MyJob.waj` job and does not send any email notification when the job is done:

```
WebWorks.AutoMap.exe --nonotify "c:\Jobs\MyJob.waj"
```

Running a Job and Deploying to a Clean Location

The following example runs the `c:\Jobs\MyJob.waj` job and deletes all the files in the deployment location before deploying the newly-generated files:

```
WebWorks.AutoMap.exe --cleandeploy "c:\Jobs\MyJob.waj"
```

Running a Job Without Deploying the Content

The following example runs the `c:\Jobs\MyJob.waj` job and does not deploy the generated output files. This example also uses the `C:\Temp\Stage` folder as the staging folder when running the job:

```
WebWorks.AutoMap.exe --nodeploy -s "C:\Temp\Stage" "c:\Jobs\MyJob.waj"
```

The staging folder provides a working folder for processing the job. ePublisher AutoMap stores the automatically generated ePublisher project, intermediate data files, and the output files in this folder.

12

Understanding How ePublisher Works

ePublisher provides the complete framework needed to solve an unlimited number of publishing issues. ePublisher provides a flexible processing model that allows you and third-party developers to customize the ePublisher workflow as needed. This workflow uses the open, standards-based XSL transformation approach to give you the flexibility and extensibility you need without locking your data in a proprietary format.

Understanding Terminology

To understand how ePublisher works, you should first understand some terminology. The following list defines several important product terms:

input formats

The types of source documents ePublisher processes, such as Adobe FrameMaker, Microsoft Word, and DITA-compliant XML files.

output formats

The types of output ePublisher can generate from your source documents. Each output format is a set of individual XSL transforms that process source documents to create the output. Quadralay provides many default output formats, such as WebWorks Help, HTML Help, Eclipse Help, Oracle Help, Sun JavaHelp, XML+XSL, Wiki Markup, Dynamic HTML, and many others. You can also create custom output formats. An output format is broken down into pipelines and stages.

format files

The files that define the output format and transforms. These files are stored in the `Formats` folder in the installation folder. You can override files to customize the transformation process and these override files are stored in a matching location within the project folder. When you save a Stationery, the override files are stored with the Stationery.

Stationery

A complete set of processing rules and styles that define all aspects of the output. The Stationery can define multiple targets, and each target can be the same or different output format. Writers use the Stationery created by the Stationery designer when they create projects and generate output. Writers can also override some default settings in their projects, such as variable values and conditions.

project

Identifies the Stationery to use, the source documents to process, and the output format and target settings to override when generating output. A project can include multiple targets.

target

Defines a deliverable for the project, based on an output format. A project can have multiple targets and each target can be the same or different output format. All targets in a project share the Style Designer properties and options, but each target has its own target settings, such as variable values and company information.

stage

The smallest discrete action possible in an output format, such as an XSL transform. ePublisher transforms source documents to a target by breaking the process into a series of steps, also known as stages. A stage is part of the transformation process that performs a specific action in the process. Stages are grouped into pipelines of related stages. In the future, non-XSL actions may also be possible. All stages define an output file type and zero or more input file types.

pipeline

A set of stages that are run in sequence. Pipelines can have dependencies on other pipelines, which can ensure that required input files are created before a pipeline runs.

Understanding the Processing Workflow

A **Stationery designer** creates Stationery that identifies the supported output formats and defines the appearance and behavior of the output generated with projects based on that Stationery. A **writer** creates a project based on the Stationery and identifies the source documents to include in the project. Then, the writer uses the project to generate output for each target defined in the project. A **developer** can create custom behaviors and formats for Stationery designers to enable and use in the ePublisher Designer console.

ePublisher uses stages, pipelines, and the `format.wffmt` file for each output format to structure and manage the processing workflow. A stage in a pipeline runs only once. The stage itself must determine the number of files to process and the number of files to emit. A stage can pass any message to another stage as long as it is emitted to a file first, and then the type and path of the file are emitted in the XML result.

ePublisher processes files as follows:

1. Apply conditions, cross-reference formats, and variables to source documents.
2. Export source documents to WIF.
3. Determine the execution order for format pipelines.
4. Select the next pipeline available for processing and execute all the stages defined in that pipeline.

To fill gaps left by XSL, ePublisher provides several XSL extension objects to support specific actions, such as processing image files, modifying the file system, and writing multiple documents from a single XSL transform.

Understanding the Transformation Process

The first pipeline in the process prepares the source documents to be transformed by XSL. ePublisher uses XSL to transform documents to output formats. ePublisher also provides extensions to XSL to support image processing and other source document operations that otherwise would require additional technologies, such as FrameScript or VBA. ePublisher requires native access to source documents, a standard starting point to start XSL transformation, and a standard method for coordinating XSL transforms on files. This first pipeline applies conditions and variables, extracts native drawings and images, and exports the source documents to the WebWorks Intermediate Format (WIF), a WebWorks XML language that enables XSL to process the source documents in later stages.

When XSL processing begins, ePublisher processes files based on type rather than by name. Each stage defines an `.xsl` file that creates a portion of the output target. When every stage in every pipeline is finished, the transformation process is complete.

Using the transformation to WIF gives you the ultimate flexibility with multiple input and output format support. Each input format, such as Microsoft Word and Adobe FrameMaker, can be transformed to any and all output formats, such as Microsoft HTML Help and WebWorks Help. As new output formats become available, all input formats are automatically supported. In addition, a new input format automatically supports all output formats.

Adapters Transform Source Documents to WIF

To extract content from source documents, ePublisher defines an adapter interface. This adapter interface allows ePublisher to transparently support a variety of source documents, such as Microsoft Word, Adobe FrameMaker, and DITA.

When processing content, ePublisher is not aware of the type of adapter in use. ePublisher simply asks for an adapter that can process a source document, and then it sends requests to the adapter associated with that document type.

Adapters handle the following tasks:

1. Reporting book files.
2. Scanning for styles, conditions, cross-references, and variables.
3. Applying specified conditions, cross-reference formats, and variables to source documents.
4. Extracting native drawings and images.
5. Exporting source documents to WIF.

This final step is where the source document is effectively brought into the world of XML/XSL processing.

Understanding WebWorks Intermediate Format (WIF)

WIF is a Quadralay Corporation standard used as an intermediate format. Quadralay chose to establish WIF rather than use another XML format because other XML formats do not provide all the functionality needed to solve conversion-related issues. Other XML standards are great for authoring structured content, but they deliberately exclude formatting information and they are not designed to define source that lacks structure. To address many conversion-related issues, such as unstructured Microsoft Word source documents with one style (Normal) used throughout the documents and customized as needed, Quadralay needed a more flexible and powerful intermediate format.

The most flexible way to resolve these issues was to create WIF, which provides a stable XML schema that can grow and change as needed for future input and output format requirements. WIF is designed to address the following goals:

- Represent source documents in a standard schema for XSL processing.
- Preserve individual word processor features with high fidelity.
- Strongly favor XSL processing over ease of authoring.

Processing Files by Type

Once source documents are transformed into WIF, XSL processing can begin. The next question is about how to organize the XSL processing. ePublisher gives you the flexibility you need to define a workflow without the drudgery. While some processes and standards require intimate knowledge of an XSL transform's input and output *file names*, ePublisher uses knowledge of an XSL transform's input and output *file types*. This distinction is subtle but powerful.

Consider processing XSL with exact file names:

- `xslt doctoc.xsl alpha.xml > alpha_toc.xml`
- `xslt doctoc.xsl delta.xml > delta_toc.xml`
- `xslt grouptoc.xsl alpha_toc.xml delta_toc.xml > group_toc.xml`

Now consider using file types in place of file names:

- `xslt doctoc.xsl Source > Doc_TOC`
- `xslt grouptoc.xsl Doc_TOC > Group_TOC`

Using file types in place of file names simplifies the specification, where the file types are defined as follows:

File Type	File Names
<i>Source</i>	alpha.xml delta.xml
<i>Doc_TOC</i>	alpha_toc.xml delta_toc.xml
<i>Group_TOC</i>	group_toc.xml

With the file type approach, you need to specify `xslt doctoc.xsl Source > Doc_TOC` only once. This specification runs for every *Source* type document found and generates the corresponding *Doc_TOC* output files.

Identifying Files to Process

Processing files based on type rather than individual file names allows developers to define workflows for any number of input and output files. To identify the files each XSL transform should process or create, ePublisher defines an XML schema to store file information. A simplified form of this schema looks like the following example:

```
<Files>
<File name="alpha.xml" type="Source" />
<File name="delta.xml" type="Source" />
</Files>
```

This XML document is fed as input to each XSL transform, and every XSL transform emits the list of generated files using this same schema. Therefore, running `doctoc.xsl` with the previous example input file list returns the following output:

```
<Files>
<File name="alpha_toc.xml" type="Doc_TOC" />
<File name="delta_toc.xml" type="Doc_TOC" />
</Files>
```

In this example, the final transform, `grouptoc.xsl`, runs with the following input file list:

```
<Files>
<File name="alpha.xml" type="Source" />
<File name="delta.xml" type="Source" />
<File name="alpha_toc.xml" type="Doc_TOC" />
<File name="delta_toc.xml" type="Doc_TOC" />
</Files>
```

This final transform returns the following output:

```
<Files>
<File name="group_toc.xml" type="Group_TOC" />
</Files>
```

The *Source* documents were fed into the `grouptoc.xsl` transform because all previous output files are available to all downstream XSL transforms. ePublisher allows developers to define as many XSL stages as needed, and each stage can process, reprocess, examine, and manage, any preceding file generated.

TOC Processing Example

The previous sections describe a TOC processing example to illustrate how ePublisher processes files. The following example shows the corresponding `format.wwfmt` file for this TOC processing example:

```
<Format>

<Pipeline name="TOC">

  <Depends pipeline="Locale" />

  <Stage type="xsl" action="doc_toc.xsl">

    <Parameter name="ParameterDependsType" value="Source" />

    <Parameter name="ParameterType" value="Doc_TOC" />

  </Stage>

  <Stage type="xsl" action="group_toc.xsl">

    <Parameter name="ParameterDependsType" value="Doc_TOC" />

    <Parameter name="ParameterType" value="Group_TOC" />

  </Stage>

</Pipeline>

</Format>
```

This representation defines the XSL transforms and file types to process.

Understanding Stationery, Projects, and Overrides

The following sections provide an overview of the ePublisher files used in the transform process. You can customize these files and store them in an override location to customize how ePublisher transforms your content. For more information about terms used in the following sections, see “Understanding Terminology”.

File Locations

ePublisher Designer allows you to create Stationery that writers can base their projects on. The Stationery stores the style settings and customized files used to define how ePublisher transforms your content. When writers create projects based on Stationery, their projects copy the customized files and settings from the Stationery and use those customized files when processing the content included in their projects. In this way, customized files in the projects override the default ePublisher files. The Stationery also allows you to quickly roll out changes to your customized processes and settings. Once you update your Stationery, writers are notified when they open a project based on the Stationery to synchronize with the Stationery, which incorporates your latest changes.

ePublisher uses file and folder locations to provide a structure where you can create and store override files. An **override file** is a customized file stored in a parallel location in a project or Stationery that is used to process the source documents instead of the default ePublisher file stored in the installation folders. These files customize how ePublisher transforms your content. ePublisher uses the following locations to store its files:

Your project folders

By default, each project is saved in the `My Documents\componentname Projects\projectname` folder, where *componentname* is the name of the ePublisher component used to create the project, such as `ePublisher Designer` or `ePublisher Express`, and *projectname* is the name of the project itself.

ePublisher installation folder

By default, ePublisher components are installed in the `Program Files\WebWorks` folder. The default transformation files are stored with ePublisher Designer in the `Program Files\WebWorks\ePublisher\ePublisher Designer` folder.

When you generate output for an output format, ePublisher first checks the project folders for the files required to complete the task. If the files are not found in the project folders, ePublisher checks the installation folders. This process allows you to override any default file in the installation folder hierarchy for one or more output formats by placing a customized file with the same name at the correct location in the project folder hierarchy.

Note: Do not modify files in the installation folders. Store customized files only in the project folder hierarchy.

The files used to transform a source document are located in several main folders in the ePublisher installation folder hierarchy:

Formats

Contains format-specific XSL files. The default files that you can customize and store in your project folder hierarchy are stored in this folder. The `Formats\Shared` folder contains XSL files used by multiple formats.

Helpers

Contains command-line programs to perform specific actions, such as compiling HTML Help `.chm` files or generating WebWorks Help search indices.

File Processing

ePublisher divides the transform process into a series of pipelines, or groups of similar stages. Each stage defines an `.xsl` file to run that creates a portion of the target. ePublisher uses a combination of the format pipelines, the `files.info` GlobalFiles record file, XSLT parameters, and the root match template in a given XSL file to perform the action identified in a stage. These stages are defined in the `format.wwfmt` file located in each format folder. The `format.wwfmt` file defines all the pipelines and stages necessary to create a specific output format, and the relationships that each of these pipelines and stages have to one another. There is no preconceived order in which pipelines run. ePublisher calculates at run time the order in which each pipeline runs, based on pipeline dependencies.

ePublisher processes files based on type rather than by name. Each stage defines an `.xsl` file to run that creates a portion of the target. All ePublisher XSL style sheets define the following global parameters by default:

Parameter	Description
GlobalFiles	Provides the <code>files.info</code> file for the project, which includes all previously generated files in the files XML schema.
GlobalProject	Provides the <code>.wep</code> project file, which defines the project as XML.
GlobalPipelineName	Specifies the name of the pipeline in which the current stage is defined.
GlobalInput	Identifies the files selected in Document Manager. This parameter provides all previously generated files derived from the <code>Generate Selected</code> command in the files XML schema.

In addition, ePublisher passes the XSL style sheets all parameters defined for the current stage in the `format.wwfmt` file. Parameters passed to each XSLT file are usually used to load the various XML node sets needed for the current stage. For example, a stage may need to take information recorded in the **Behaviors** pipeline and merge it with information in the **Links** pipeline. The location of information generated in each stage is stored in the `files.info` file. Future stages can use the `files.info` file to learn the location of previously generated information needed to complete that particular stage. When that stage finishes, it notifies the `files.info` file that the information it created is available for future stages to use, if necessary.

Processing files based on type rather than by individual names allows you to define workflows for any number of input and output files. This flexibility allows you to define as many XSL stages as you need, and each stage can use any preceding files generated. You can create or delete pipelines, add or delete stages, or insert elements that dictate when to run a stage or pipeline.

ePublisher File Types

This section provides specific information about the files most commonly used to customize ePublisher and its transformation processes.

Format Trait Info (*.fti) Files

Format Trait Info files have the `.fti` file extension. Format Trait Info files are located in both the ePublisher `Formats\Shared` folders, and the `Format` folders of each target.

In every project, the user has a series of format and style options to customize the appearance of his output. In the ePublisher consoles, these options are accessed and manipulated through target settings and Style Designer. For ePublisher, these customization options are defined in the Format Trait Info files. Manipulating these files allows you to customize the location of options, create or delete options and parameters, adjust their valid values, or choose new default values.

For ePublisher to process a Format Trait Info file, the `.fti` file must have the same file name, not including the `.fti` file extension, as an XSL file in the current format folder hierarchy. For example, ePublisher processes the table of contents using the `toc.fti` and `toc.xsl` files in the same folder. The information displayed in ePublisher represents all Format Trait Info files associated with a specific format. Two or more Format Trait Info files may define the same `<Setting />` element, but the ePublisher console displays that setting only once.

When you generate output, ePublisher reviews the settings specified in target settings and Style Designer, stores the settings in the `.wep` project file, and incorporates the settings in the generated output.

format.wwfmt Files

A `format.wwfmt` file is stored in each of the specific format folders. The `format.wwfmt` file defines all the stages required to create output, and the relationships between each of these stages. These stages are grouped in pipelines of related stages. When every stage of every pipeline has finished, ePublisher is done generating the output. There is no preconceived order in which pipelines run. ePublisher calculates at run time the order in which each pipeline runs. You can create or delete pipelines, add or delete stages, or insert a `<Depends />` element that dictates when a stage or pipeline runs.

files.info Files

ePublisher creates and stores the `files.info` file for each target with the project files. Each stage completes a small portion of the transformation. A completed stage creates new information that contributes to the appearance or functionality of the transformed content. When a stage finishes, ePublisher writes the location of the processed information in the `files.info` file for use by other stages. Each stage can use information in the `files.info` file and record information in the `files.info` file for use by another stage.

Stationery Design Project .wep File

The Stationery design project .wep file contains all the configuration information required to define the Stationery and generate output. This file includes the sample source document names, settings, options, and customizations. You can use ePublisher Designer to modify a Stationery design project, and to save it as Stationery.

Project .wrp File

The project `.wrp` file contains all the information required to generate output. This file includes all the source document names, settings, options, and customizations.

Stationery .wxsp File

A Stationery .wxsp file is based on settings and options defined in a Stationery design project and saved as Stationery. By saving a project as Stationery, all the settings and customizations that you captured in your project, including project format overrides, are automatically implemented in any new projects based on the Stationery.

XSL Match Templates

Style sheets are made of a number of templates, each of which defines what the XSLT processor should do when it matches a particular node in the XML source document. The XSLT processor populates the result document by instantiating a sequence of templates. Instantiation of a template means that the XSLT processor performs the following tasks:

- Copies any literal data from the template to the target
- Executes the XSLT instructions in the template

Templates are defined using the `<xsl: template>` element. The `match` attribute in the `<xsl:template>` element indicates which parts of the source document should be processed with the particular template.

Root Match Templates

Each XML document has a single root element. This element encloses all the following elements and is therefore the parent element to all the other elements. When the XSLT processor applies a style sheet to an XML document, it begins processing with the root element of the XML source document. To process the root element, the XSLT processor searches the style sheet for a template rule that matches the root element. A template rule matches the root element when the value of the `template match` attribute is `/` (slash).

If the user explicitly defines a template rule that matches the root element, the XSLT processor finds it and applies that template to the entire XML document. If the XSLT processor does not find an explicitly defined template rule that matches the root element, the processor implements the default template that matches the root element. Every style sheet includes this default template.

Root Match Templates in ePublisher

The root match template has a number of responsibilities in ePublisher:

- Recording files and dependencies
- Loading node sets
- Progress recording for the extension object that lives in the `wwprogress` namespace.
- Up-to-date checking on the projects output files.
- Writing output

Extension Objects

While extremely powerful, XSL has shortcomings when used to perform system-level scripting tasks. Some operations, such as working with files and advanced string operations, are not included in the standard XSL language.

XSL provides a generic extension mechanism to add new features to the base language. ePublisher provides a standard set of extension objects that enable XSL to generate all the required formats.

XSL extensions live in a specific namespace. You can define your own prefix to associate with a given namespace, but using a consistent naming convention makes life easier.

Output and Console Customizations

Presenting information in print can involve very different presentation decisions than displaying information in an online, Web-friendly format. The appearance of the online content can vary from one platform to another, and one online format may have features that are not available in another online format.

In a transform performed by ePublisher, you can make several customization choices through various options in the ePublisher console. For example, target settings allow you to specify whether company information is displayed on the page, and where it is located on the page. Style Designer allows you to customize any style in the document to appear a certain way. By creating specific rules and conditions for how the document should appear when the transform is complete, you can add functionality or information for specific audiences, or implement specific features of a given format.

By using XSL, ePublisher allows you to customize the process even further. You can make additions, deletions, or modifications to the XSL files in ePublisher to further customize the output for your specific needs. You can even add, remove, or modify the options available in target settings and Style Designer.

Project Format and Target Overrides

ePublisher gives you complete control over the final output. However, some project modifications cannot be made through the ePublisher console. In these instances, you may need to override the default XSL files used by ePublisher to achieve the results you need.

When ePublisher generates output for the first time for a project, ePublisher reviews the `format.wxfmt` file in the folder for the appropriate format. This file tracks the actions required to generate the desired format. If there are no user customizations, all the files referenced by the `format.wxfmt` file are in the Format or Transforms folders in the installation folder hierarchy. By default, ePublisher first checks the project folder hierarchy for each required file before getting the files from the installation folder. With this process, you can override one or more default files by creating the same folder hierarchy in a project folder and storing a customized file with the same name at the correct location in the project folder hierarchy.

Note: You can override default files for all targets of a specific output format type in the project. You can also override default files for a specific target in a project without affecting other targets of the same output format type in that project. Overrides for a target override any customizations you specified in the override folders for a format.

Creating Format Overrides

This section defines how to override a default file for all targets of a specific output format type in the project. Do not modify the files in the installation folder hierarchy. These files are the default files and should remain as is. These files are also overwritten when you install new ePublisher releases. Instead, store and incorporate your customized files with your Stationery so your projects based on the Stationery get those customizations.

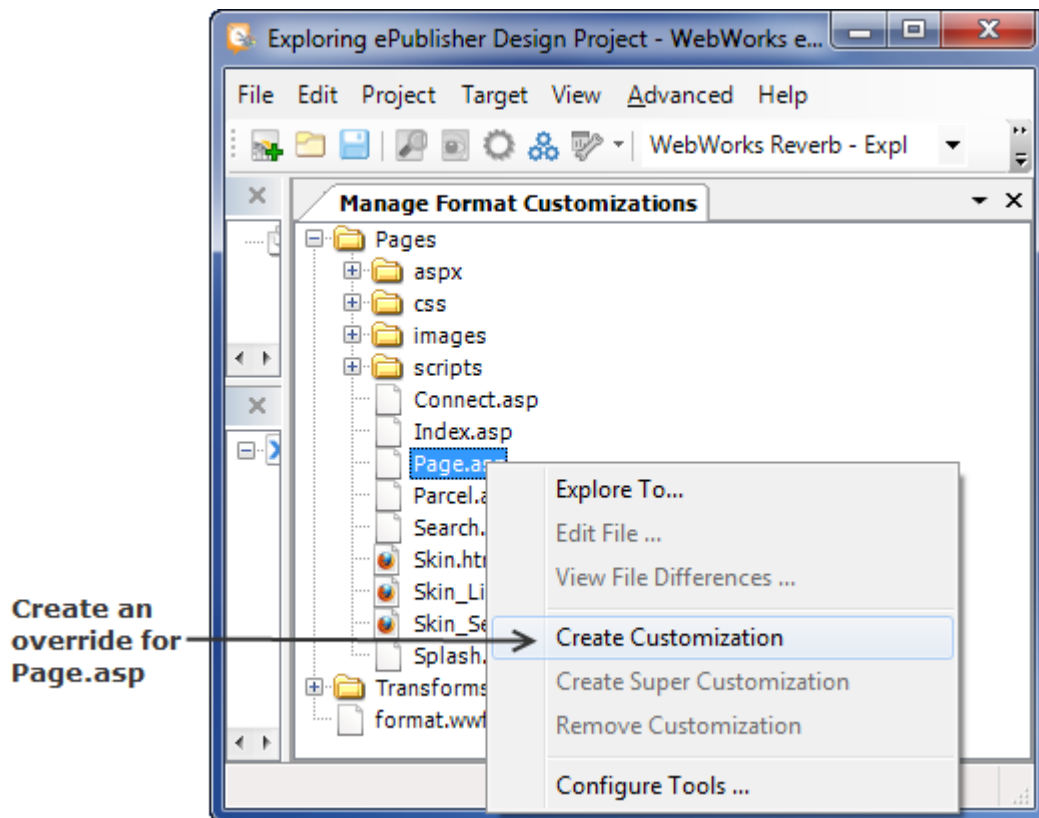
Note: Changes to `.xsl` and Format Trait Info `.fti` files are considered advanced customizations and are not supported by the Quadralay Product Support department. If you are familiar with XSL, you may find that customizations to `.xsl` and `.fti` files are a powerful way to achieve very specific results in your generated output or output deployment. However, you are responsible for maintaining any `.xsl` or `.fti` file overrides that you implement. The Quadralay Product Support department does not provide support for these advanced customizations.

File names used in ePublisher are case sensitive. Make sure the file and folder names you create exactly match the default file and folder names in the installation folder hierarchy. Do not copy any files into the project folder hierarchy that you are not overriding.

The following task assumes that ePublisher is installed in the default location and your ePublisher projects are stored in the `My Documents` folder. If you installed ePublisher to a non-default location, or if you store your projects in a folder other than the `My Documents` folder, adjust the paths in the following task.

To override a project format

1. In your Stationery design project, on the **Advanced** menu, click **Manage Format Customizations**.
2. In the displayed collection of folders and files, navigate to the file that you wish to override and highlight it.
3. To create an override for the selected file, right click and select the **Create Customization** menu. This will create an exact copy of the original file and place it into the appropriate location within the project directory. Now you can safely modify this file as desired as well as viewing the file differences between your override and the original.



4. Use either of the right-click mouse menu items: **Explore To...** or **Edit File ...** to access and modify your file override.

The next time you generate your project, the modified file automatically overrides the default file and the changes you made are incorporated into the output for all targets that produce that output format.

Note: When you save the ePublisher project as Stationery, the project format overrides you have created are saved with your Stationery. This Stationery can then be used to create future projects in ePublisher Express and ePublisher AutoMap.

Creating Target Overrides

This section defines how to override default files for a specific target in a project without affecting other targets of the same output format type in that project. Do not modify the files in the installation folder hierarchy. These files are the default files and should remain as is. These files are also overwritten when you install new ePublisher releases. Instead, store and incorporate your customized files with your Stationery so your projects based on the Stationery get those customizations.

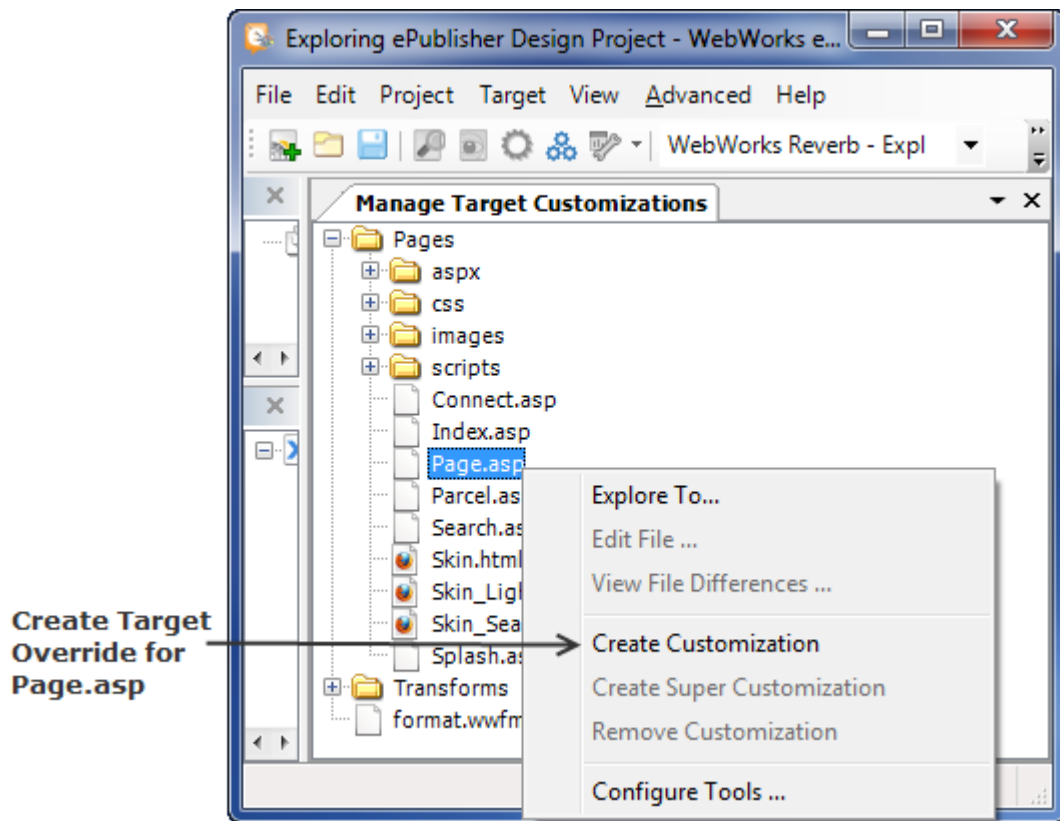
Note: Changes to `.xsl` and Format Trait Info `.fti` files are considered advanced customizations and are not supported by the Quadralay Product Support department. If you are familiar with XSL, you may find that customizations to `.xsl` and `.fti` files are a powerful way to achieve very specific results in your generated output or output deployment. However, you are responsible for maintaining any `.xsl` or `.fti` file overrides that you implement. The Quadralay Product Support department does not provide support for these advanced customizations.

Make sure the file and folder names you create exactly match the default file and folder names in the installation folder hierarchy. Do not copy any files into the project folder hierarchy that you are not overriding.

The following task assumes that ePublisher is installed in the default location and your ePublisher projects are stored in the `My Documents` folder. If you installed ePublisher to a non-default location, or if you store your projects in a folder other than the `My Documents` folder, adjust the paths in the following task.

To override a project target

1. In your Stationery design project, on the **Advanced** menu, click **Manage Target Customizations**.
2. In the displayed collection of folders and files, navigate to the file that you wish to override and highlight it.
3. To create an override for the selected file, right click and select the **Create Customization** menu. This will create an exact copy of the original file and place it into the appropriate location within the project directory. Now you can safely modify this file as desired as well as viewing the file differences between your override and the original.



4. Use either of the right-click mouse menu items: **Explore To...** or **Edit File ...** to access and modify your file override.

The next time you generate your project, the modified file automatically overrides the default file and the changes you made are incorporated into the output for that target. These customizations also override any override files you saved in the project folder hierarchy for the associated output format.

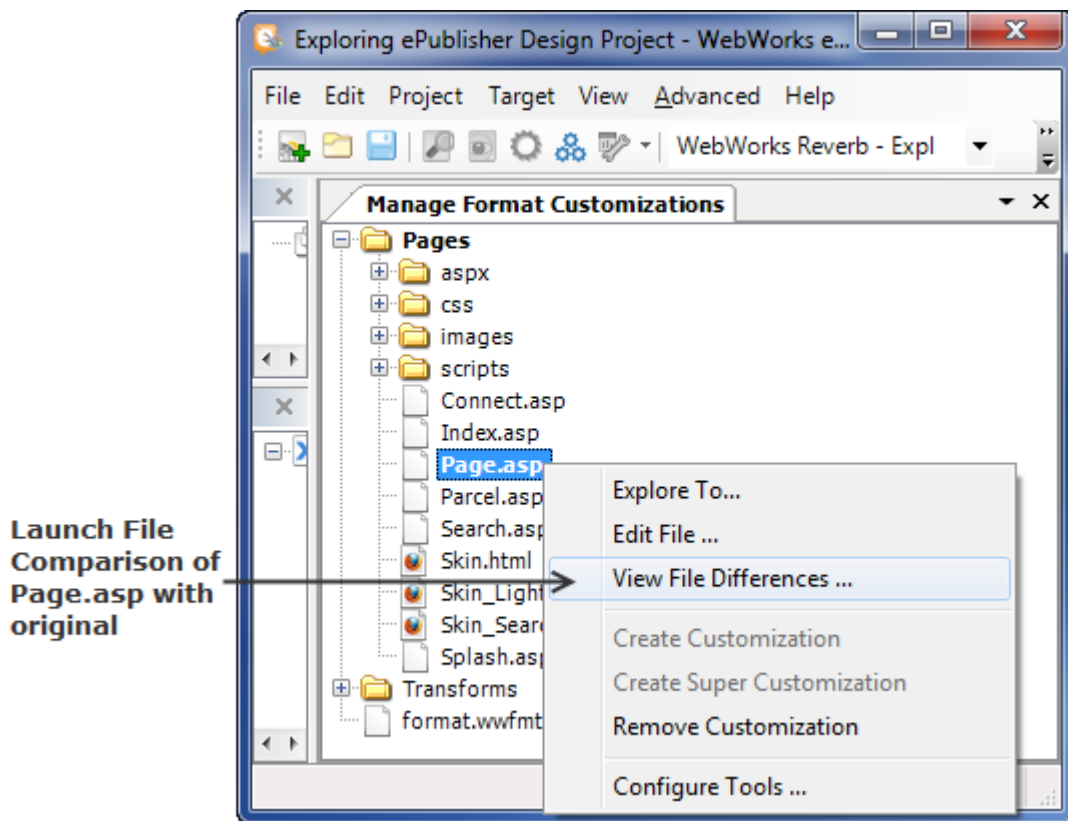
Note: When you save the ePublisher project as Stationery, the project target overrides you have created are saved with your Stationery. This Stationery can then be used to create future projects in ePublisher Express.

Managing Format/Target Overrides

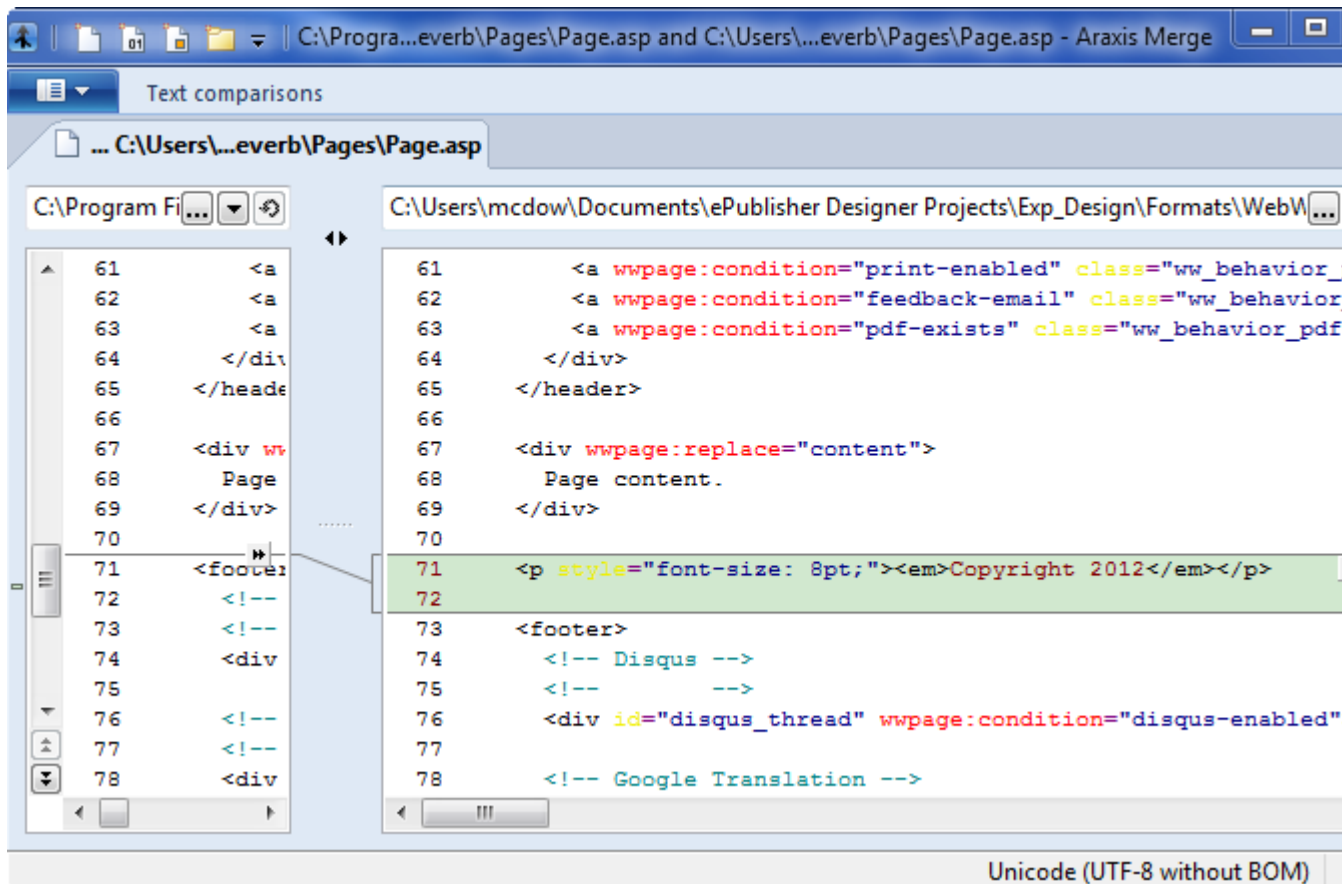
ePublisher assists you in the management of your project's overrides by graphically highlighting files that have been overridden as well as highlighting folders that contain files that have been overridden. In addition, you can also have ePublisher launch a 3rd-party *Diff* (also called a line-comparison) tool that will intuitively display what modifications have been made in your overridden file.

To View File Differences

1. In your Stationery design project, on the **Advanced** menu, click **Manage Format Customizations** (or Target for target overrides).
2. In the displayed collection of folders and files, navigate to the overridden file that you wish to compare and highlight it.
3. To compare the differences of the selected override, right click and select the **View File Differences...** menu. This will launch your configured *Diff* program and automatically display the differences between your file and the original.



4. You can examine how your override is different from the original. In addition, you can modify your overridden file directly from within the *Diff* tool.



Creating Super Overrides

An XSL super override is simply a way for the stationery designer to add an XSL override to a file in the Transforms directory without having to copy the entire template structure of the file.

Note: Support XSL overrides is only available through Study Hall and not Standard Support. For more information regarding Study Hall, please go to http://www.webworks.com/eschool/study_hall/.

Here is an example from the WebWorks wiki of XSL that you can use as a starting point for learning about this feature:

```
<!-- Override the mode wwdoc:Table from the base content.xml -->

<!-- -->

<xsl:template match="wwdoc:Table" mode="wwmode:content">

<xsl:param name="ParamTable" select="." />

<!-- Parameters passed in to this processing context. We need to pass them along. -->

<!-- -->

<xsl:param name="ParamSplits" />

<xsl:param name="ParamCargo" />

<xsl:param name="ParamLinks" />
```

```
<xsl:param name="ParamTOCData" />
```

```
<xsl:param name="ParamSplit" />
```

```
<!-- Manually set all table formatting in this template. -->
```

```
<!-- -->
```

```
<html:table cellpadding="0" cellspacing="5" border="1" style="margin-top: 0.6em; margin-bottom: 0.6em;">
```

```
<xsl:for-each select="$ParamTable/wwdoc:Caption/wwdoc:Paragraph[1] | $ParamTable/preceding-sibling::wwdoc:Paragraph[@style = 'Caption']" />
```

```
<html:caption>
```

```
<html:p>
```

```
<html:b>
```

```
<xsl:for-each select="./wwdoc:TextRun/wwdoc:Text">
```

```
<xsl:value-of select="@value" />
```

```
</xsl:for-each>
```

```
</html:b>
```

```
</html:p>
```

```
</html:caption>
```

```
</xsl:for-each>
```

```
<xsl:for-each select="$ParamTable/wwdoc:TableHead | $ParamTable/wwdoc:TableBody | $ParamTable/wwdoc:TableFoot">
```

```
<xsl:variable name="VarSection" select="." />
```

```
<xsl:variable name="VarTagName">
```

```
<xsl:choose>
```

```
<xsl:when test="local-name($VarSection) = 'TableHead'">
```

```
<xsl:text>thead</xsl:text>
```

```
</xsl:when>
```

```
<xsl:when test="local-name($VarSection) = 'TableBody'">
```

```
<xsl:text>tbody</xsl:text>
```

```
</xsl:when>
```

```

<xsl:when test="local-name($VarSection) = 'TableFoot'">
<xsl:text>tfoot</xsl:text>
</xsl:when>
</xsl:choose>
</xsl:variable>

<xsl:element name="{ $VarTagName}" namespace="'http://www.w3.org/1999/xhtml'">
<xsl:for-each select="$VarSection/wwdoc:TableRow">
<xsl:variable name="VarRow" select="." />

<html:tr>
<xsl:for-each select="$VarRow/wwdoc:TableCell">
<xsl:variable name="VarCell" select="." />

<html:td>
<!-- Pass control back to the base processing flow. -->
<!-- -->
<xsl:apply-templates select="$VarCell/wwdoc:Paragraph" mode="wwmode:content">
<xsl:with-param name="ParamSplits" select="$ParamSplits" />
<xsl:with-param name="ParamCargo" select="$ParamCargo" />
<xsl:with-param name="ParamLinks" select="$ParamLinks" />
<xsl:with-param name="ParamTOCData" select="$ParamTOCData" />
<xsl:with-param name="ParamSplit" select="$ParamSplit" />
</xsl:apply-templates>
</html:td>
</xsl:for-each>
</html:tr>
</xsl:for-each>
</xsl:element>
</xsl:for-each>

```

```
</html:table>
```

```
</xsl:template>
```

To use the super override

1. In your Stationery design project, on the **Advanced** menu, click **Manage Format Customizations** (or Target for target overrides)
2. In the displayed collection of folders and files navigate to the **Transforms** directory, and click on the XSL file you wish to modify
3. Right click the XSL file, and click Create Super Customization
4. In the text editor, you will see the XSL that indicates:

```
<!-- Write templates here -->
```

```
<!-- -->
```

5. Add the custom XSL, please note that the code will have to reference the original XSL file in order for ePublisher to process it correctly. For more information, please refer to this [wiki page](#).
6. Save XSL override, save ePublisher project, reopen and generate output for ePublisher to process this override

Customizing Page Templates

Page templates are files that form the skeleton structure of generated output files. Often users will create overrides for these files so that they can precisely control the look and function of generated output files. The most common type of page templates to customize are the `Page.asp` and `Splash.asp` files, which are used in most all of the output format types.

Namespace and Attributes

ePublisher defines the namespace: `wwpage` for page templates as follows:

```
xmlns:wwpage="urn:WebWorks-Page-Template-Schema"
```

The following shows the page template attributes that can be used along with their purpose.

Attribute	Purpose
<code>wwpage:condition</code>	Controls when elements should be preserved in the generated output.
<code>wwpage:content</code>	Use this attribute to replace the <code>innerHTML</code> or <code>innerHTML</code> of an element.
<code>wwpage:replace</code>	Use this attribute to replace the <code>innerHTML/innerHTML</code> as well as the <code>outerHTML/outerXML</code> .
<code>wwpage:attribute-<name></code>	Use this attribute to emit a generated attribute value with name: <code>name</code> . In addition, you can specify a value of either: <code>relative-to-output</code> or <code>copy-relative-to-output</code> or <code>wwsetting:<target-setting-name></code> . Specifying the value using <code>relative-to-output</code> is useful for file type attributes that require a pathname relative to the parent file. If the <code>copy</code> version is used, it additionally copies the file into the correct location within the generated output directory.
<code>wwpage:NoBreak</code>	Use this attribute to collapse all white space between two elements into nothing.
<code>wwpage:Import</code>	Use this attribute to indicate an that an import operation will take place using one of the <code>*-from-*</code> <code>wwpage</code> attributes.
<code>wwpage:content-from-file</code>	Use this attribute to import the contents of the specified file as the <code>innerHTML</code> of an element.
<code>wwpage:replace-from-file</code>	Use this attribute to import the contents of the specified file as the <code>innerHTML</code> as well as the <code>outerHTML</code> of an element.
<code>wwpage:content-from-lookup</code>	Use this attribute to import the contents of the specified item name generated by ePublisher as the <code>innerHTML</code> of an element.
<code>wwpage:replace-from-lookup</code>	Use this attribute to import the contents of the specified item name generated by ePublisher as the <code>innerHTML</code> as well as the <code>outerHTML</code> of an element.

Note: If you are using `wwpage:attribute-<name>` to express attribute that uses a namespace of its own, you can use a “-” character in place of the “:” character. For example: `wwpage:attribute-xml-lang` is used to generate the `xml:lang` attribute in the output.

The following shows a typical usage of `wwpage` attributes within a `Page.asp` or `Splash.asp` page template.

Attribute	Example usage in Page.asp page template
<code>wwpage:condition</code>	<code><hr wwpage:condition="header-exists" align="left" /></code>

Attribute	Example usage in Page.asp page template
wwpage:content	<code><title wwpage:content="title">Title</title></code>
wwpage:replace	<code><div wwpage:replace="content">Page content.</div></code>
wwpage:attribute-<name>	<code><link rel="StyleSheet" href="css/print.css" wwpage:attribute-href="copy-relative-to-output" type="text/css" media="print" /></code>
wwpage:NoBreak	<code> <wwpage:NoBreak /> </code>
wwpage:Import	<code><wwpage:Import wwpage:replace-from-file="scripts/common.js" /></code>
wwpage:content-from-file	<code><div wwpage:Import wwpage:content-from-file="scripts/common.js"></div></code>
wwpage:replace-from-file	<code><wwpage:Import wwpage:replace-from-file="css/webworks.css" /></code>
wwpage:content-from-lookup	<code><div wwpage:Import wwpage:content-from-lookup="catalog-css"></div></code>
wwpage:replace-from-lookup	<code><wwpage:Import wwpage:replace-from-lookup="catalog-css" /></code>

Logical AND/OR in condition attributes

The `wwpage:condition` attribute supports both logical AND as well as OR to provide for multiple conditions in a single attribute expression. For a logical AND, use white space as a separator between conditions. For a logical OR, use a single comma as a separator between conditions. The following shows a few simple examples of using multiple conditions:

Logic Type	Example
AND (use space)	<code><hr wwpage:condition="header-exists company-email-exists" /></code>
OR (use ",")	<code><hr wwpage:condition="header-exists, footer-exists" /></code>
AND with OR	<code><hr wwpage:condition="header-exists company-email-exists, footer-exists company-email-exists" /></code>

Working with URL and File Paths in Page Templates

The `wwpage:attribute-<name>` attribute can be used to specify several behaviors when working with URL and File paths. The following shows the list of available modifiers for this `wwpage: attribute`.

Attribute Modifier	Example
relative-to-output	<code><script type="text/javascript" language="JavaScript1.2" src="wwhdata/common/context.js" wwpage:attribute-src="relative-to-output"></script></code>
copy-relative-to-output	<code><script type="text/javascript" src="scripts/common.js" wwpage:attribute-src="copy-relative-to-output"></script></code>

Attribute Modifier	Example
absolute-to-output	<code><external-graphic content-height="scale-to-fit" width="1in" height="1in" src="url('wwformat:Pages/images/Logo.png') " wwpage:attribute-src="absolute-to-output" /></code>
resolved-uri	<code><external-graphic src="url({wwformat:Pages/images/rms_doc_logo.png}) " wwpage:attribute-src="resolved-uri" content-width="100px" content-height="100px" left="0pt" top="0pt" /></code>
resolved-path	<code><external-graphic src="{wwformat:Pages/images/rms_doc_logo.png}" wwpage:attribute-src="resolved-path" content-width="100px" content-height="100px" left="0pt" top="0pt" /></code>

Using Replacement Types ({}) in Page Templates

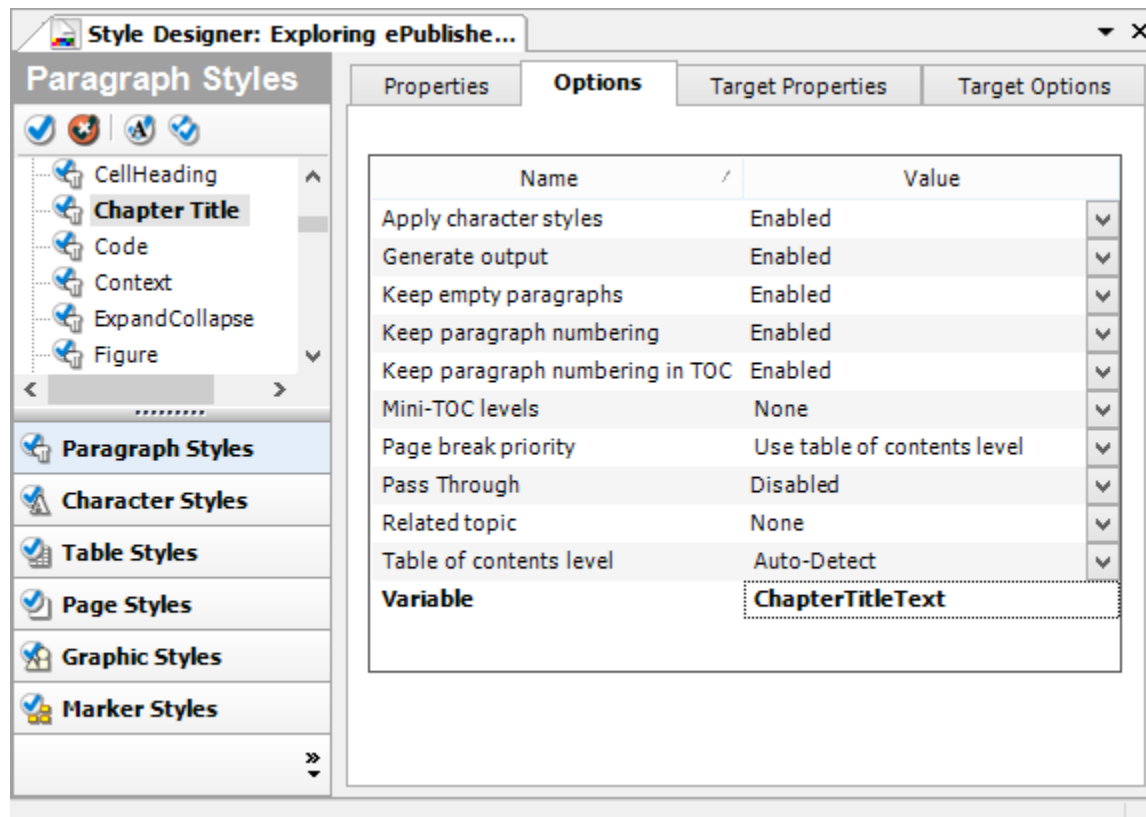
In page templates, you can use the “{}” characters to indicate a parameter to use in an attribute replacement. The “{}” characters are used to embed a path or parameter within an attribute value. For example, the following is a typical replacement for an HTML `img` tag, demonstrating how “{}” characters can be used to improve the result of the replacement.

```
.
```

Using ePubublisher Style Variables in Page Templates

A powerful feature for capturing text in your source content based on their assigned style name is available by using the `wwvars:<Variable_Name>` in your page template file

To capture text from your source documents, you first need to configure the **Variable** style option within your ePubublisher Designer project. The **Variable** style option is available for Paragraph, Character, and Marker styles.



In the ePubublisher Style Designer, select the Variable option and assign it a value that will be used in your page template.

Once you have assigned a variable name to the style, any time ePubublisher encounters that style it will set the value of that variable to the text of that style in the source content.

To access the variable in your page template, you would use the `wwvars:<Variable_Name>` attribute, similar to the following.

```
<div wwpage:content="wwvars:ChapterTitleText">
Chapter Title Appears Here
</div>
```

13

Features - From Input to Output

This is a quick reference guide for using ePublisher

Popups

A popup window is a window that is smaller than standard windows and typically does not contain some of the standard window features such as tool bars or status bars. Popup windows display when users hover over or click on a link. The popup window closes automatically as soon as the users click somewhere else

Using Paragraph Styles

Input Instructions

“Using Paragraph Styles to Create Popup Windows in Word”

“Using Paragraph Formats to Create Popup Windows in FrameMaker”

“Using Paragraph Styles to Create Popups in DITA Source Documents”

ePublisher Instructions

“Using Paragraph Styles To Create Popup Windows”

Using Markers

Input Instructions

“Using Markers to Create Popup Windows in Word”

“Using Markers to Create Popup Windows in FrameMaker”

ePublisher Instructions

“Using Marker Styles to Create Popup Windows”

Available Outputs

- “WebWorks Help”,
- “Microsoft HTML Help”
- “Sun JavaHelp”
- “Oracle Help”

Related Topic

Related topics provide a list of other topics that may be of interest to the user of the current topic. For example, you could have a section called Creating Web Pages in your help. You may also have many other topics, such as HTML Tags and Cascading Style Sheets, that relate to creating Web pages. Identifying these related topics for users can help them find the information they need and identify additional topics to consider. However, providing these types of links as cross references within the content itself may not be the most efficient way to present the information

Using Paragraph Styles

You can use a paragraph style to define a related topics list. Create a unique paragraph style to use specifically for the related topics list items. The writer should create a list of links in a topic in the source document, and apply the paragraph style to each list item.

Input Instructions

“Creating Related Topics in Word”

“Creating Related Topics in FrameMaker”

“Creating Related Topics in DITA Source Documents”

ePublisher Instructions

“Using a Paragraph Style for Related Topics Lists”

Available Outputs

- “Simple HTML” on page 66
- “Dynamic HTML”
- “WebWorks Help”
- “Microsoft HTML Help”
- “Sun JavaHelp”, “Oracle Help”
- “Wiki Markup”

See Also Links

See Also links are associative links (Alinks) that identify other topics that may be of interest to the user of the current topic. These links use internal identifiers to define the links and the list is built dynamically based on the topics available when the user displays the links. See Also links are important to use with larger help sets and merged help sets

Using Markers

Input Instructions

- “Creating See Also Links in Word”
 - Microsoft Help Instructions: More Info...
- “Creating See Also Links in FrameMaker”
 - Microsoft Help Instructions: More Info...

ePublisher Instructions

“Enabling See Also Functionality”

Available Outputs

- “WebWorks Help”
- “Microsoft HTML Help”

Context Sensitive Help

Context sensitive help is simply a way to identify a topic by using a standard location combined with a topic identifier. Certain formats, such as WebWorks Help 5.0 and WebWorks Reverb, further require a content identifier to be used in conjunction with a topic identifier. Context-sensitive help links provide content based on the context of what the user is doing. In many cases, this help content is based on the window that is open and active. For example, the **Help** button on a window in a software product can open a specific help topic that provides important information about the window:

- What the window allows you to do
- Brief concepts needed to understand the window
- Guidance for how to use the window
- Descriptions about each field on the window, valid values, and related fields
- Links to related topics, such as concepts and tasks related to the window

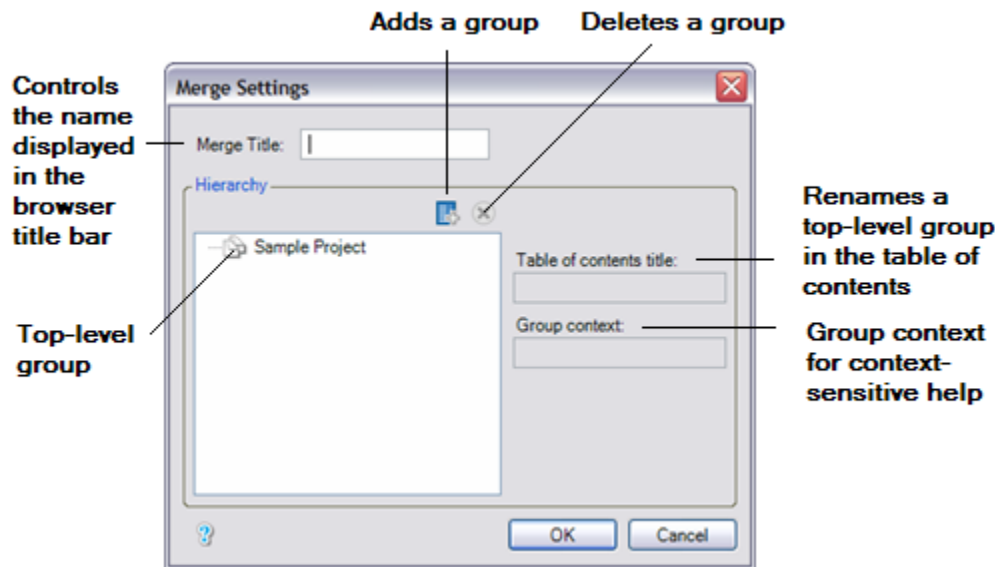
Using Markers

Input Instructions

- “Creating Context-Sensitive Help in Word”
- “Creating Context-Sensitive Help in FrameMaker”
- “Creating Context-Sensitive Help in DITA Source Documents”

ePublisher Instructions

Context sensitive help takes values from the Merge Settings window:



Note: The group context must be unique. The group context is used to refer to help sets in WebWorks Help and WebWorks Reverb.

To initially identify groups, refer to the **Document Manager** for the top level groups, and then you can rename groups as needed through the **Merge Settings** dialog. This is important for context sensitive help because it will give you the basis for your group context in the merge settings and help you organize the topic identifiers within each group context. Each top-level group is considered its own help set. For example, the Microsoft HTML Help format will create a new CHM file per top-level group in the **Document Manager**.

- “Defining Filename Markers for Context-Sensitive Help Links”
- “Defining TopicAlias Markers for Context-Sensitive Help Links”

Available Outputs

- “WebWorks Help”
 - More Info...
- “Microsoft HTML Help”
 - More Info...
- “Sun JavaHelp”
 - More Info...
- “Oracle Help”
 - More Info...
- “Eclipse Help”
- “WinHelp” on page 92,

Multi-volume Help

Merged help, which is sometimes also referred to as multivolume help, is a help system with a single set of files created from output from multiple groups from within a project. Merged help takes the table of contents, index, and search data from each top-level group entry-point file and combines this information to create a single, consolidated help system

Using Groups

- “Organizing Source Documents Using Groups”

ePublisher Instructions

- “Creating Top-Level Groups”

Available Outputs

- “WebWorks Help”,
- “Microsoft HTML Help”,
- “Eclipse Help”

Mini TOC

A miniature table of contents (mini-TOC), also known as a partial table of contents, provides a list of the topics in the upcoming section. The topic titles are displayed as links beneath the current topic heading for easier navigation within the section. By default, ePublisher does not create a mini-TOC in topics within your output. You can configure ePublisher to generate mini-TOCs and you can define your mini-TOC levels in Style Designer.

Using Paragraph Styles

Input Instructions

- “Paragraph Styles in Word”
- “Paragraph Formats in FrameMaker”
- “DITA Specialization”

ePublisher Instructions

“Creating a Miniature Table of Contents”

Available Outputs

- “Simple HTML” on page 66,
- “Dynamic HTML”
- “WebWorks Help”
- “Microsoft HTML Help”
- “Eclipse Help”, “WinHelp” on page 92
- “Sun JavaHelp”, “Oracle Help”
- “Wiki - Confluence”
- “Wiki - MediaWiki”
- “Wiki - MoinMoin”

Custom TOC icon

For some output formats, ePublisher allows you to add a marker to a heading that defines the icon to use for the table of contents entry for that heading. For example, you can use a unique icon for new topics, or for specific types of information. ePublisher provides the TOCIconWWHelp, TOCIconHTMLHelp, TOCIconJavaHelp, and TOCIconOracleHelp markers

Using Makers

Input Instructions

- “Customizing Table of Contents Icons in Word”
 - Microsoft HTML Help: More Info...
 - WebWorks Help: More Info...
 - Java Help/Oracle Help: More Info...
- “Customizing Table of Contents Icons in FrameMaker”
 - Microsoft HTML Help: More Info...
 - WebWorks Help: More Info...
 - Java Help/Oracle Help: More Info...
- “Customizing Table of Contents Icons for Topics in DITA Source Documents”
 - Microsoft HTML Help: More Info...
 - WebWorks Help: More Info...
 - JavaHelp/Oracle Help: More Info...

ePublisher Instructions

- “Using Custom Icons on the Contents Tab in WebWorks Help”

Available Outputs

- “WebWorks Help”,
- “Microsoft HTML Help”,
- “Sun JavaHelp”,
- “Oracle Help”,

Wiki Categories and Labels

On Wikis, **categories**, which are referred to as **labels** in some Wiki formats, are used to organize Wiki content. Categories help group together pages that have similar subjects.

MoinMoin and Media Wiki use the term *category* to describe page grouping functionality. Confluence uses the term *label* to describe page grouping functionality.

More Info...

Using Markers

Input Instructions

- “Specifying Wiki Categories or Labels in Word”
- “Specifying Wiki Categories or Labels in FrameMaker”
- “Specifying Wiki Categories or Labels in DITA”

Available Outputs

- “Wiki - Confluence”
- “Wiki - MediaWiki”
- “Wiki - MoinMoin”

Page Styles

In ePublisher, the overall appearance of each topic of your output is controlled by page styles. You can modify page style properties with Style Designer to define whether to include navigation browse buttons on each page, whether to include company logo and contact information, and other design elements for each page that uses a specific page style.

Using Markers

Input instructions

- “Assigning Custom Page Styles in Word”
- “Assigning Custom Page Styles in FrameMaker”
- “Assigning Custom Page Styles to Pages in DITA Source Documents”

ePublisher instructions

- “Defining the Appearance of Pages”

Available Outputs

- Page Styles are available in all formats
 - Note: Not all options are supported in PDF Format
- Note: PDF-XSL Format supports additional properties
 - Master Page: More Info...
 - Pagination: More Info...