

Corda Builder User Guide

Version 7.0



Corda Technologies, Inc.

350 South 400 West, Suite 100

Lindon, UT 84042

Headquarters: (801) 805-9400

Fax: (801) 805-9405

Sales: (801) 805-9500

Technical Support: (801) 805-9505

Press Contact: (801) 805-9431

Sales Email: sales@corda.com

Support Email: support@corda.com

Corda Server and Corda Builder at run-time use FreeLib.jar. FreeLib.jar contains the run-time library, the source code, and the license information for all classes and methods it contains. Portions of FreeLib.jar contain software Copyright (C) 1998,2000 Steady State Software Ltd. All rights reserved.(<http://www.steadystate.com/>) Released under the GNU Lesser General Public License. Portions of FreeLib.jar contain software developed by Shazron Abdullah (shazron@stormgate.com) in Java, original code Copyright (C) 2001 Opaque Industries - <http://www.opaque.net/>. Released under the GNU Lesser General Public License. Portions of FreeLib.jar contain software Copyright (C) 1999,2000, 2001, 2002 by Bruno Lowagie, Copyright (C) 2000, 2001, 2002 Paulo Soares, and others. All rights reserved. (<http://www.lowagie.com/iText/>) Released under the GNU Lesser General Public License. This product (FreeLib.jar) includes software developed by the Apache Software Foundation (<http://www.apache.org/>)

Microsoft, Windows, and PowerPoint are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Screen shots reprinted by permission from Microsoft Corporation. Adobe, Acrobat, Illustrator, and Acrobat Reader are registered trademarks of Adobe Systems Incorporated in the United States and/or other countries. Macromedia, ColdFusion, and Flash are trademarks or registered trademarks of Macromedia, Inc. in the United States and/or other countries. Sun, Java, and Javascript are trademarks of Sun Microsystems, Inc. in the United States and other countries. Netscape and Netscape Navigator are registered trademark of Netscape Communications Corporation in the U.S. and other countries. SVG is a trademark of the World Wide Web Consortium. UNIX is a trademark registered in the United States and other countries, licensed exclusively through X/Open Company, Ltd. Linux is a registered trademark of Linus Torvalds. All other trademarks are the property of their respective owners.

Copyright © 1996-2004 CORDA Technologies Inc.™ - Corda™ - Corda Builder™ - PopChart™ - OptiMap™ - Highwire™ - All Rights Reserved.

www.corda.com



Contents

CHAPTER 1 Preface

About Corda Builder	1-2
Getting Help	1-3
Documentation Conventions	1-6
Where To Go Next	1-8

CHAPTER 2 Corda Builder Overview

About Image Template Files	2-2
Corda Builder	2-5
Corda 7 Object Reference	2-18

CHAPTER 3 Using Corda Builder

Working with Objects	3-2
The Corda 7 Gallery	3-8
Creating a Basic Graph	3-12
Creating a Basic Map	3-16
Troubleshooting Corda Builder	3-19

CHAPTER 4 Graphs

General Graph Configuration	4-3
Graph Data	4-6
Bar-Based Graphs	4-13
Pie Graphs	4-18

- CONTENTS
- Scales and Grids
-
-

Line-Based and Plot Graphs	4-20
Heat Maps	4-24
Gauges	4-25

CHAPTER 5 Scales and Grids

Terminology	5-3
Scale Types	5-5
Value Scales	5-6
Category Scales	5-17
Synchronizing Scales	5-21
Grids	5-22
Tick Marks	5-26
Scale Markers	5-27
Scale Titles	5-30

CHAPTER 6 Maps

About Maps	6-2
Creating Maps	6-10
Map Data	6-14
Customizing Maps	6-16

CHAPTER 7 Map Ranges

Range Generation	7-2
Range Calculation	7-5
Range Names	7-8
Range Settings	7-9

CHAPTER 8 Other Corda Objects

Legends	8-2
Text Boxes	8-5
Shapes	8-6
Images	8-21

CHAPTER 9 Drilldown Effects

Viewing Drilldown	9-3
Identifying Drilldown	9-4
Graph Drilldown	9-5
Map Drilldown	9-7
Other Object Drilldown	9-9
Drilldown Meta Tags	9-10
Javascript Drilldown	9-11

CHAPTER 10 Data Annotations

Data Labels	10-2
Hover Text	10-10
Graph Notes	10-17

CHAPTER 11 Additional Information

Hover Objects	11-2
Descriptive Text	11-5
Data Tables	11-11
User Preferences	11-14
Numeric Format Settings	11-20
Multiple Graphs and Maps in the Same Project	11-21
Creating Templates	11-24
Animations	11-25

- CONTENTS
- *Image Formats*
-
-

Creating New Color Themes	11-26
Dynamic Objects	11-29
Corda Fonts	11-32

CHAPTER 12 **Image Formats**

Macromedia Flash	12-2
SVG	12-7
PNG	12-9
JPEG	12-11
PDF	12-13
EPS	12-17
TIFF	12-20
EMF	12-22
Comparison of Image Format Features	12-24
Table of MIME Types	12-25

CHAPTER 13 **Publishing the Project**

Publishing a Project	13-2
Accessing Published Files	13-5
Output Features	13-7
Publishing Static Images on the Web	13-10
Publishing Dynamic Image Template Files with Corda Builder's Sample Code	13-19
Previewing a Corda Image	13-23
Where Do I Get the SVG or FLASH Plug-In?	13-24
How Do I Print a Corda Image?	13-25
Tips for Creating Good Image Template Files	13-26

PREFACE

Welcome to the *Corda® Builder™ User Guide*. This guide helps you learn how to use Corda Builder to design images for use with Corda Server, as well as building Image Template files for dynamic graph and map creation. It also helps you publish static graphs and maps.

This chapter includes the following sections:

- [About Corda Builder](#)
- [Getting Help](#)
- [Documentation Conventions](#)
- [Where To Go Next](#)

Included in Corda 7 is Corda Technologies' suite of server-based data presentation tools. This suite comprises three separate modules: PopChart®, OptiMap™, and Highwire™, all of which use Corda Builder to produce dynamic, data-driven images and documents.

These modules allow you to translate data into state-of-the-art Corda images—eye-catching, high-resolution, and interactive data-driven images—for incorporating into Corda Server images.

- PREFACE

- *About Corda Builder*
-
-

ABOUT CORDA BUILDER

Corda Builder is a graphical design tool that runs on most desktop computers. With an intuitive interface, Corda Builder helps designers create graph or map Image Template files for Corda images. These Image Template files can be saved or uploaded to Corda Server and used to generate dynamic Corda images.

Images created by Corda Builder can contain a variety of graphs and maps, fed with on-demand dynamic data. They can include explanatory text boxes, notes, or hover text that appears as a viewer rolls over certain parts of the image. They can even include interactive drilldown effects, such as linking to another graph or map when a user selects a certain data item, or executing a custom Javascript™ function.

Corda Builder can create graph or map images in up to eight different formats, including Macromedia® FLASH™, SVG™, PNG, JPEG, PDF, EPS, TIFF, and EMF. For visually impaired users, it also automatically generates a 508-compliant, [d](#) link text description of the graph or map.

Corda Builder also includes features for developers, including ITXML and PCScript editors, that provide precise control over graph and map content, and sample code generation to help you create Web pages to contain the Corda images.

Finally, non-technical users can use Corda Builder as a standalone tool for publishing graphs and maps in static Web pages or for converting Web pages to PDF documents.

Note: *Although Corda Builder is installed with Corda 7, it must be purchased separately. To enforce this restriction, Corda Builder uses a different license key than Corda 7.*

GETTING HELP

For assistance using Corda Builder, use the following resources:

- [Documentation](#)
- [OnLine Resources](#)
- [Corda Support](#)

DOCUMENTATION

Corda 7 documentation is available in both PDF (which is great for printing) and HTML (which is great for browsing and searching) formats. Access Corda 7 documentation from the [Documentation](#) shortcut or by clicking [g Help > Documentation](#) in Corda Builder. Documentation files are stored in `<product_root>\Resources\docs`.

This documentation is updated as performance is enhanced and new features are added. Visit the Corda Technologies Web site (<http://www.corda.com>) regularly for the latest documentation.

This Guide is just one of several books documenting the features and use of Corda 7. The following documentation is included with Corda 7:

CORDA INSTALL AND ADMINISTRATION

This guide provides an overview of Corda 7, and instructions for installing and configuring Corda 7. It is intended primarily for Corda 7 administrators.

CORDA 7 DEVELOPER REFERENCE

This guide provides developer-level information related to PCScript, ITXML, Corda 7 server commands, and the Corda Embedder. It also provides instructions and samples related to using these programming resources dynamically with Corda Server.

CORDA 7 HIGHWIRE GUIDE

This guide teaches you how to translate Web pages and other HTML or XHTML documents into PDF documents using Corda Server's Highwire module.

CORDA 7 MAP GUIDE

This guide contains a catalog of the various map types available with Corda 7, along with installation instructions and usage information.

CORDA 7 GRAPH GUIDE

This guide contains instructions on using all of the various graph types available with Corda 7. It also contains example graphs, data, and Image Template files.

- PREFACE
- Getting Help
-
-

CORDA BUILDER USER GUIDE

This guide contains detailed instructions for designing graphs or maps with Corda Builder. When you click the **Help** button in Corda Builder, a context-sensitive version of this guide pops up. It is intended for Corda designers.

ONLINE RESOURCES

The Corda Technologies Web site (<http://www.corda.com>) has a variety of resources to help you use Corda Builder, including the following:

Online documentation Our most current documentation can always be found on the Web at <http://www.corda.com/devzone/docs>.

Product Updates Patches and updates for Corda Technologies products are at <http://www.corda.com/download>. Please look for updates and patches before contacting Corda technical support.

Corda Builder in Action A number of Corda Builder demonstrations are running on the Corda Technologies Web site at <http://www.corda.com/examples>. These demonstrations can help illustrate using Corda Builder within the organization.

Code Examples Many code examples are online at <http://www.corda.com/devzone/code>. These help you perform common tasks like connecting to a database in various Web environments.

Developer Zone The Corda developer zone is a collection of developer resources, including a knowledge base and searchable forums for discussing Corda 7 implementations with other developers. It is located at <http://www.corda.com/devzone>.

CORDA SUPPORT

If you cannot find what you're looking for elsewhere, contact Corda technical support by email at support@corda.com or by phone at (801) 805-9505 (Monday through Friday, 8 a.m. to 5 p.m. Mountain Standard Time). For complete support policies, see <http://www.corda.com/support>.

Note: *Users of evaluation versions of Corda 7 are not entitled to Corda support.*

DOCUMENTATION CONVENTIONS

To improve readability, this guide uses some special conventions when referring to files, URLs, example code or text, and options or buttons in dialogs and menus. For the most part, these should be pretty intuitive, but we mention them here just in case.

DIALOG/MENU NAMES AND OPTIONS

Names of menus, dialogs, and options are in green with a slightly larger arial font (e.g., **Data Editor**).

For brevity, whenever you have to go through multiple layers of menus or dialogs to arrive at the desired menu, dialog, or option, a greater than > sign is used to indicate that the next menu, dialog, or option can be found under the previous.

For example, instead of saying, “Select the **Save** option from the **File** pull-down menu,” this documentation says, “Select **File > Save**.” Similarly, instead of saying, “Change the value of the **Port** option in the **Address/Port** screen of the **Settings** section in the Administration Console,” this documentation says, “Change the **Settings > Address/Port > Port** option in the Administration Console.”

EXAMPLE CODE

Small segments of code (including HTML and XML) that appear in the body of a normal paragraph are shown in courier font, and are colored light green or yellow (e.g., `graph.Transposed(true)`).

Medium-sized segments of code appear on a single indented line in a courier font. For example:

```

```

Long segments of code look similar, but are on multiple lines and are delimited with a thin horizontal line above and below the code. An example number appears at the top left corner of the segment for reference. The code segment is also titled. [Example 1.1](#) below demonstrates a longer segment of code.

- PREFACE

- *Documentation Conventions*
-
-

Example 1.1 Example of Long Code Segment

```
myImage = new CordaEmbedder();
myImage.imageTemplate("examples\test.itxml");
myImage.pcScript("graph.setCategories(Jan;Feb;Mar;Apr;May;Jun)grap
                h.setSeries(Temperature;30;40;50;60;70;80)graph.addh
                overText(1,1,Look at the pretty pictures)");
myImage.getEmbeddingHTML();
```

EXAMPLE TEXT

Any other example text, including text that is typed into a dialog window and values of attributes or variables, appear italicized in a slightly larger arial font (e.g., *false* in “The default value of `graph.Transposed` is *false*”).

FILE NAMES AND TRADEMARKED NAMES

File names and trademarked names are shown in gray with a slightly larger arial font (e.g., `myfile.txt`).

File names are described using the `<product_root>` variable, which varies from system to system. It is the folder into which you installed Corda 7. The default `<product_root>` is `C:\Corda\Corda7` on Microsoft Windows® systems or `/usr/local/Corda/Corda7` on Linux/UNIX® systems.

Additionally, this documentation uses the `<document_root>` variable to indicate the default path where Corda 7 saves, or looks for, Corda images and Image Template files. By default, it is the only location from which Corda Server can load files. The default `<document_root>` is `<product_root>\Resources\image_templates`.

URLS

URLs are always shown in either blue or gray with a slightly larger arial font. A blue URL indicates an active, clickable link (e.g., <http://www.corda.com>). A gray URL link indicates an inactive, example link (e.g., `http://www.yourserver.com`).

If the example URL is in reference to Corda Server, make it a valid URL by replacing `www.yourserver.com` with the address (and port) of Corda Server. Although this guide does not require any previous knowledge of Corda Server, it is useful to know a little about how graphs and maps can be embedded in Web pages with Corda Server. For more

information, see [“Corda Server” on page 1-5](#) of the *Corda 7 Install and Administration Guide*.

- **PREFACE**

- *Where To Go Next*

-
-
-
-
-

WHERE TO GO NEXT

This chapter has introduced Corda Builder and its design interface, which offers enormous flexibility for Image Template design. The remaining chapters in this guide provide additional detail on the many features and options for customizing graphs and maps using Corda Builder.

These chapters provide an overview of Corda Builder:

- Chapter 1, “[Preface](#)”
- Chapter 2, “[Corda Builder Overview](#)”
- Chapter 3, “[Using Corda Builder](#)”

These chapters discuss working with graphs:

- Chapter 4, “[Graphs](#)”
- Chapter 5, “[Scales and Grids](#)”

These chapters discuss working with Maps:

- Chapter 6, “[Maps](#)”
- Chapter 7, “[Map Ranges](#)”

This chapter discusses working with other Corda Builder objects:

- Chapter 8, “[Other Corda Objects](#)”

These chapters discuss additional features that can be used to enhance data accessibility in Corda images and make working with Corda Builder easier:

- Chapter 9, “[Drilldown Effects](#)”
- Chapter 10, “[Data Annotations](#)”
- Chapter 11, “[Additional Information](#)”

Finally, these chapters discuss issues related to publishing Corda images to the Web:

- Chapter 12, “[Image Formats](#)”
- Chapter 13, “[Publishing the Project](#)”

CORDA BUILDER OVERVIEW

This chapter provides an overview of the major components of Corda® Builder™ and the tools available for designing Image Template files. The information in this chapter is organized into the following sections:

- [About Image Template Files](#)
- [Corda Builder](#)
- [Corda 7 Object Reference](#)



- **CORDA BUILDER OVERVIEW**

- *About Image Template Files*

ABOUT IMAGE TEMPLATE FILES

Corda Builder saves Corda Builder graph and map information in project files, also referred to as *Image Template* files. Image Template files allow you to customize and publish graphs and maps again and again, as needed.

Create an Image Template file from scratch, or customize one from a pre-designed template using the **Graph Wizard**. Corda Builder saves Image Template files in a text-based XML format (using an `.itxml` extension) that is easy to manipulate in a text editor, and it is flexible enough to modify or create Image Template file on the fly.

Corda Builder displays sample data in Image Template files to show what graphs and maps might look like, but Corda Server typically applies live data to the Image Template files dynamically when they are published. Although the data changes, Corda Server retains the layout and formatting options specified for the Image Template file.

Note: *Because Image Template files are merely templates for graphs and maps, it is hard to know how the resulting graph looks when fed real data. For tips on creating an Image Template file that is effective with dynamic data, see [“Tips for Creating Good Image Template Files” on page 13-26.](#)*

By default, Corda Builder saves Image Template files in `<product_root>\Resources\image_templates`, which is accessible from the **Document Root** shortcut. Several sample Image Template files are stored in this folder.

OPENING IMAGE TEMPLATE FILES

Corda Builder opens a blank Image Template file when it loads. To open an existing Image Template file, do one of the following:

OPEN USING FILE MENU. Select **File > Open** and enter the project’s filename, or browse to the desired file.

OPEN USING RECENT PROJECTS Select **File > Recent Image Templates** and select the project.

OPEN USING MICROSOFT WINDOWS START MENU Access the Image Template file directory by selecting the **Start > All Programs > Corda 7 > Image Templates**. Double-click any Image Template file to open it in Corda Builder.

OPEN WITH CORDA 7 GALLERY

Click one of the Gallery buttons to open the Corda 7 Gallery; then locate and open the desired Image Template file. For more information about using Corda 7 Gallery, see [“The Corda 7 Gallery” on page 3-8.](#)

SAVING IMAGE TEMPLATE FILES

Save an Image Template file using one of the following options:

- **Save:** Use **File > Save** if the Image Template file has been previously saved. Corda Builder replaces the previously saved version of the file. If the file has not been previously saved, it prompts you for a file name and location.
- **Save As.** Use **File > Save As** to save the file for the first time, or to save a new copy of an existing file. **Save As** prompts you for a file name and location.

Note: *Because Corda Server looks for Image Template files in its document root directory, save projects that you intend to use as Image Template files for Corda 7 in `<product_root>\Resources\image_templates` (or in an `image_templates` subdirectory).*

IMAGE TEMPLATE FILE CONFIGURATION

Corda Builder provides broad project-level configuration options for the Image Template file, meaning that the configurations apply to the entire Image Template file, not just a single graph or map object. It is important to make this distinction because a project can contain a large number of graph, map, and other objects. Each object can also be modified independently of all others, as described in the object-specific chapters in this guide.

Corda Builder supports the following project-level properties:

- [Project Dimensions](#)
- [Project Border](#)
- [Project Background](#)

PROJECT DIMENSIONS

To create larger or smaller output images, adjust the size of the Project Canvas. To change the size of the Project Canvas, click and drag the square “handles” on the project border, or specify specific dimensions in the project’s properties.

Attributes related to project dimensions are available in the **Layout** property in **Object Properties**. More information on these properties and attributes is available in the *Corda 7 Object Reference* (see “[Corda 7 Object Reference](#)” on page 2-18).

Note: *Changing the project size does not scale the objects within the project to the new size. Changing the size of the project might make it necessary to resize individual project objects to better fit the project space. To change an object’s size, see “[Working with Objects](#)” on page 3-2.*

- **CORDA BUILDER OVERVIEW**
- *About Image Template Files*
-
-

PROJECT BORDER

Optionally, Corda Builder can display visible project borders surrounding the Project Canvas.

Attributes related to project borders are available in the *Canvas* property in *Object Properties*. More information on these properties and attributes is available in the *Corda 7 Object Reference* (see [“Corda 7 Object Reference” on page 2-18](#)).

PROJECT BACKGROUND

Optionally, Corda Builder can provide a colored project background to help project objects stand out. The selected background color appears behind all objects in the Image Template file. Backgrounds can be a solid color or a gradient.

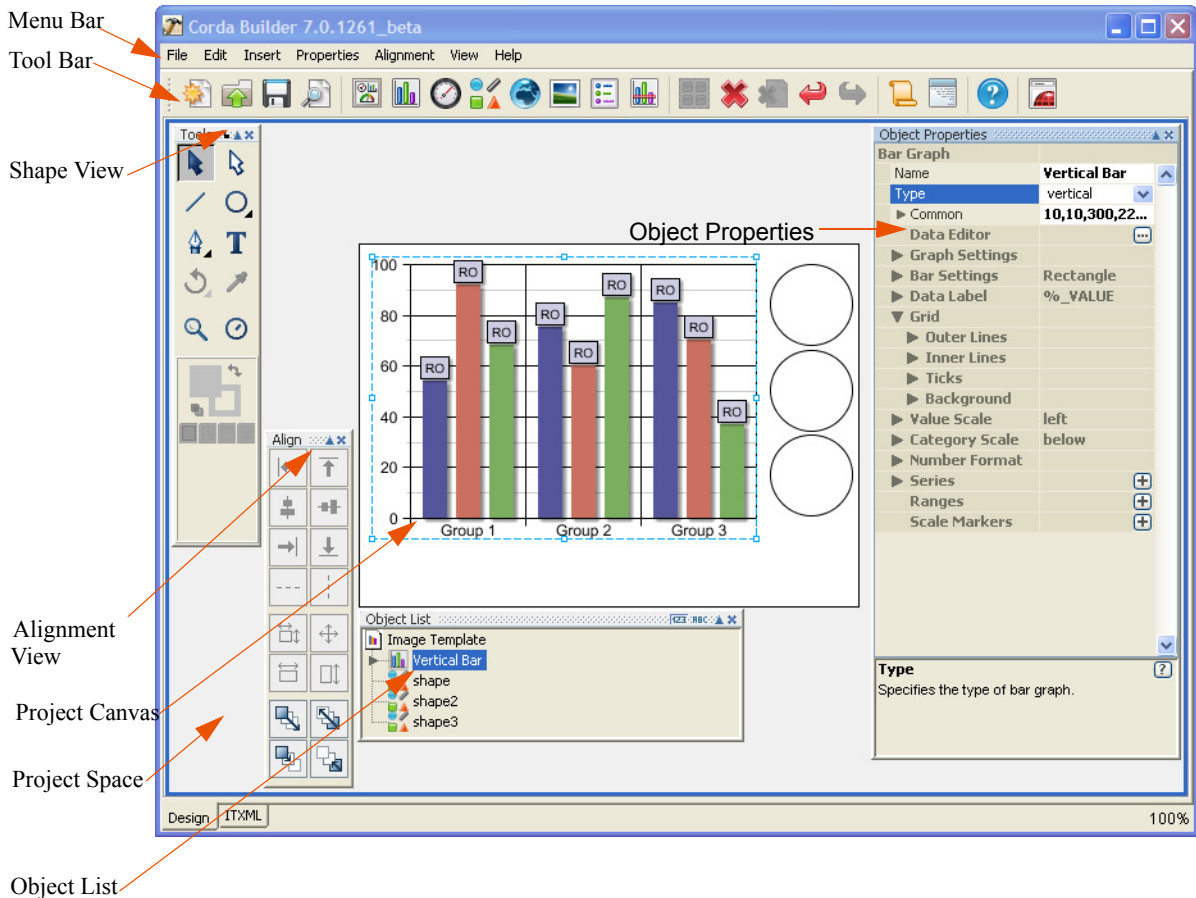
Attributes related to project background are available in the *Canvas* property in *Object Properties*. More information on these properties and attributes is available in the *Corda 7 Object Reference* (see [“Corda 7 Object Reference” on page 2-18](#)).

CORDA BUILDER

Corda Builder is a powerful graphical tool for building Corda graphs, maps, and Image Template files.

Corda Builder can be installed as part of the Corda 7 package, or separately, using the Corda Builder installer.

When creating static graphs and maps, or designing Image Template files for Corda Server (i.e., not developing the Web pages that use these Image Template files), download the Corda Builder installer. Developers creating both Image Template files and Web pages for dynamic graphs and maps using Corda Server should download the Corda 7 package.



- **CORDA BUILDER OVERVIEW**
- *Corda Builder*
-
-

INSTALLING CORDA BUILDER

To install Corda Builder, download the installer from the Corda Technologies Web site (<http://www.corda.com>), run the installer, and follow the prompts. For more information, see “[Installation Process](#)” on page 2-8 of the *Corda 7 Install and Administration Guide*.

After installation, launch Corda Builder by executing one of the following:

- Windows: Launch `<product_root>\Builder\CordaBuilder.exe`
- Linux/UNIX: Open `<product_root>\Builder` and execute `./CordaBuilder.bin`

Note: *To enable the Corda Builder Preview feature, you must first load Corda Server. For more information, see “[Project Canvas](#)” on page 2-7.*

CORDA BUILDER LAYOUT

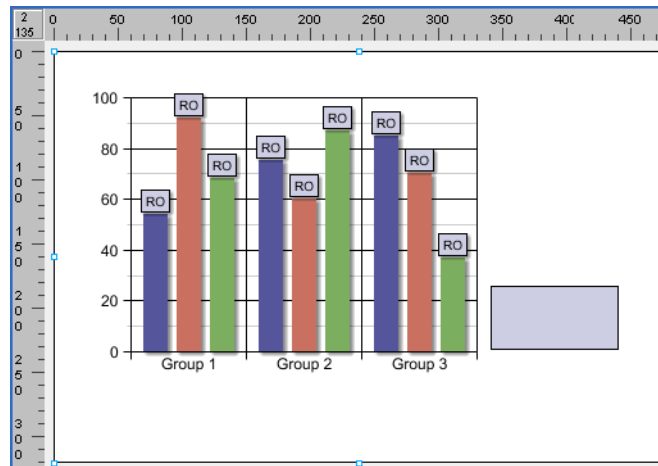
The Corda Builder interface provides the following panes to aid you in dashboard construction. Each pane is detachable to facilitate organizing your desktop as needed.

- [Project Canvas](#)
- [Object List](#)
- [Object Properties](#)
- [Shape Tools](#)
- [Alignment Tools](#)

PROJECT CANVAS

The **Project Canvas** provides a view of the current Corda Builder project. It displays all associated objects and the placement of each. The **Project Canvas** defines the area included in the final Corda image.

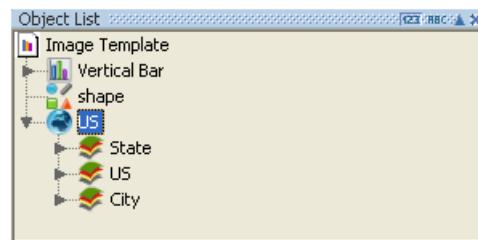
Corda Builder includes rulers along the x and y axes of the project space to help you place objects on the **Project Canvas**. The point 0,0 is placed to match the upper-left corner of the **Project Canvas**. Because of this, it is possible to have negative ruler values, noted in red on the ruler.



Note: If you have enabled *Fit in bounds*, any components that lie outside of the **Project Canvas** are repositioned inside of the **Project Canvas** when the final image is generated.

OBJECT LIST

The **Object List** provides a view of all objects used in the current project. Use **Object List** to browse to and select objects in the project.



- **CORDA BUILDER OVERVIEW**
- *Corda Builder*
-
-

Note: *The Object List view includes three small icons in the title bar: the arrow icon lets you shrink the Object List down to just the title bar, or vice versa; The “123” icon sorts the Object List by layer (back to front); and the “ABC” icon sorts the Object List by object name (A to Z).*

OBJECT PROPERTIES

The **Object Properties** pane provides a list of all properties that can be configured for the object type currently selected in the **Object List**. Use **Object Properties** to modify the property values associated with the XML objects in the project.

An object’s properties are organized in a way that reflects the underlying XML structure. Properties indicate XML tags within the object, and each property in a given group represents an attribute of the associated XML tag. Because XML is a hierarchical language, properties can be nested to represent multiple levels in the XML structure.

Within **Object Properties**, property hierarchy can be expanded or compressed as needed to make it easier to view the relevant XML settings, and work with a particular set of properties within the XML hierarchy.

Object Properties	
Map Shape	
Name	AL
▶ Common	0,0,,,true,top-left,false,true
Value	no-data
Long Name	Alabama
Code	01
Parent	US
Name Class	State
X	530.0
Y	311.0
Original Color	<input type="color" value="#CCCCFF"/>
Logical Center X	27.0
Logical Center Y	43.0
Extra Info	
▼ Shape Settings polygon	
<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> Fill Type	solid
<input type="checkbox"/> Fill Color	white
<input checked="" type="checkbox"/> Border Width	1.0
<input type="checkbox"/> Border Color	black
▶ Shadow Settings	off
<input type="checkbox"/> Description	
▶ Number Format	
▶ Label	%_NAME
▶ Hover Text	
▶ Drilldown	
Hover Items	
Fill Type	
Specifies the shape object's fill type.	

Modify a value in **Object Properties** by typing directly in the value field, or, if a specific set of values is supported, a drop-down button appears at the right end of the value field. Click the drop-down button to select from a list of supported values for that property.

CORDA BUILDER OVERVIEW

Corda Builder

Some properties provide an ellipsis [...] button that opens a separate dialog for entering a property value, particularly if the value is longer than the default value field in **Object Properties**.

Properties and attributes can also include two types of checkboxes: one that enables attribute overrides (for overriding previously set attribute values); and one that enables a given property or attribute.

Located directly below **Object Properties**, the description field gives a brief description of the currently selected XML object. Get more information about an XML object by clicking the Help (?) button located both to the right of the object name, and in the description field. This opens a description of the XML object from the *Corda 7 Object Reference* (see “[Corda 7 Object Reference](#)” on page 2-18).

Note: *The Object Properties view includes a small arrow icon in the title bar that allows you to shrink it down to just the title bar, or vice versa.*

SHAPE TOOLS

The **Shape Tools** pane provides a rich set of tools that lets you create and manipulate Shape objects of various types. For more information on using Shape Tools, see “[Shapes](#)” on page 8-6.



Note: The Tools view includes a small arrow icon in the title bar that allows you to shrink it down to just the title bar, or vice versa.

ALIGNMENT TOOLS

The **Alignment Tools** pane provides tools for aligning one or more objects in relation to another object, and moving objects within the project layers. This allows you to effectively position objects within the Project Canvas, rather than using less accurate manual positioning. For more information about Alignment options, see [“Aligning Objects” on page 3-4](#).



Note: The Align view includes a small arrow icon in the title bar that allows you to shrink it down to just the title bar, or vice versa.

- **CORDA BUILDER OVERVIEW**
- *Corda Builder*
-
-
-

MENU BAR

The following table describes all Corda Builder menu items.

FILE MENU

The File menu contains options for managing project files.

Menu Item	Functionality
New	Creates a new project from a blank canvas. Automatically closes the current document.
Open	Opens an existing project. Automatically closes the current document.
Insert File	Inserts the components of an existing project into the currently opened project.
Close	Closes the current project.
Recent Image Templates	Lists the latest (user-specified) number of projects accessed in Corda Builder. Selecting one of these projects closes the currently opened project.
Save	Saves the current project.
Save As	Saves the current project under a different name.
Save as Gallery Item	Saves the selected object as an item in the Corda 7 Gallery. For more information about the Corda 7 Gallery, see "The Corda 7 Gallery" on page 3-8 .
Save Canvas as Gallery Item	Saves the current Project Canvas configuration as an item in the Corda 7 Gallery. For more information about the Corda 7 Gallery, see "The Corda 7 Gallery" on page 3-8 .
Publish	Publishes the project as a static Web page or image.
Published Image Templates	Opens the directory in which the projects are published.
Highwire	Allows you to convert a Web document to PDF format.
Preview	Opens the user's default browser and shows how the project appears as a dynamically generated image in the Web browser.
Font Converter	Opens the Corda 7 Font Converter. For more information about the Font Converter, see "Corda Fonts" on page 11-32 .
Exit	Exits the program.

EDIT MENU

The Edit menu provides options related to modifying the current project.

Menu Item	Functionality
Undo	Undoes the last command.
Redo	Redoes the last undo command.

Menu Item	Functionality
Cut	Cuts the selected object and adds it to the clipboard.
Copy	Copies the selected object and adds it to the clipboard.
Paste	Pastes the last saved item to the clipboard.
Delete Object	Deletes the selected object.
Next Object	Selects the next object.
Previous Object	Selects the previous object.
Edit PCScript	Opens the PCScript editor. For more information about PCScript, see Appendix B, "PCScript," of the <i>Corda 7 Developer Reference</i> .
Edit Sample Code	Opens the Sample code editor, which demonstrates how to deploy the Image Template file in a variety of Web application environments and how to connect the Image Template file to dynamic data.
Preferences	Opens the User Preferences dialog. User preferences include the default configuration for the Graph Wizard , the location for template files, and the location for saved projects.

INSERT MENU

The Insert menu provides options for adding objects to the current project.

Menu Item	Functionality
Insert from Gallery	Adds a Gallery object to the project.
Insert Graph	Adds a graph to the project using the Graph Selection tool.
Insert Map	Adds a map to the project using a file.
Insert Image	Adds an image to the project.
Insert Dynamic Image	Adds a dynamic image to the project.
Insert Legend	Adds a legend object for the selected graph or map. Supports only one legend per graph or map.
Insert Scale Marker	Adds a scale marker object for the selected graph. Supports multiple scale markers per graph.

- **CORDA BUILDER OVERVIEW**
- *Corda Builder*
-
-
-

PROPERTIES MENU

The Properties menu changes, based on the item selected. For more information on this process, see [“Changing Object Properties” on page 3-3](#).

Menu Item	Functionality
Data Editor	Opens the Data Editor for the selected graph. For more information about the Data Editor, see “Enter Data” on page 3-13 .
Change Graph Type	Opens the Graph Selection dialog, from which you can change the type of the currently selected graph. This is identical to changing the graph’s Type attribute Object Properties.
Restore Original Map Size	Restores the selected map or image object to its original dimensions.
Select Layer	Allows you to select a specific map layer in the selected map object. This is identical to selecting the map layer from the Object List.
Convert Map to Shapes	Disassembles the selected map into its component shapes.
Copy Area Layer	Copies the selected map layer to the clipboard.
Add Data Point to Map	Launches the Add Data Point dialog from which you can add a new data item to the selected Points data layer in the map.
Combine into New Map	Combines the selected shapes into a new map object.
Convert to Gauge Indicator	Combines the selected shapes into a gauge indicator shape for a radial gauge. For more information, see “Radial Gauges” on page 4-30 .
Convert to Divider Object	Converts the selected object into a divider object. For more information, see “Dividing a Shape” on page 8-19 .
Convert to Polygon	Converts the selected shape object from its current type to a polygon. This lets you manipulate the shape (such as rotating it) that is not possible for some shape types.

ALIGNMENT MENU

The Alignment menu changes the alignment of the currently selected object within the **Project Canvas**. For more information on aligning objects, see [“Common Object Actions” on page 3-2](#).

Menu Item	Functionality
Align Left	Aligns the object to the left edge of another selected object.
Align Center	Centers the object within to another selected object.
Align Right	Aligns the object to the right edge of another selected object.
Align Top	Aligns the object to the top edge of another selected object.
Align Middle	Aligns the object to the middle of another selected object.
Align Bottom	Aligns the object to the bottom edge of another selected object.
Same Width	Sets the object width to that of another selected object.

Menu Item	Functionality
Same Height	Sets the object height to that of another selected object.
Overlay	Overlays the object on top of another selected object.
Move to Front	Moves the object to the foremost layer in the Corda image.
Move to Back	Moves the object to the backmost layer in the Corda image.
Move Forward One	Moves the object forward one layer in the Corda image.
Move Back One	Moves the object back one layer in the Corda image.

VIEW MENU

The View menu toggles the various Corda 7 component views for display.

Menu Item	Functionality
Align	Opens/closes the Align tool view.
Object List	Opens/closes the Object List view.
Object Properties	Opens/closes the Object Properties view.
Tools	Opens/closes the Shape Tools view.
Ruler	Enables/disables the project ruler.
Zoom	Increases/decreases magnification of the project.

HELP MENU

The Help menu provides access to Corda Builder's various help options.

Note: *You must have an installed Web browser to use the online help system.*

A PDF version of the Corda Builder documentation is provided in
<product_root>\Resources\docs.

Menu Item	Functionality
Documentation	Opens the default browser to the Corda online documentation.
Object Reference	Opens the default Web browser to the Corda 7 Object Reference Index page.
Visit Corda.com	Opens the default browser to www.corda.com .
Message from Corda	Opens a Welcome page with links to Documentation, Developer Information, Product Information, and the Corda Newsletter.
Check for Updates	Opens the default Web browser to the Corda Builder Updates page at http://www.corda.com/download/update.html .
Corda Technical Support	Opens the default Web browser to the Corda Support Web page (www.corda.com/support).













- **CORDA BUILDER OVERVIEW**
- *Corda Builder*
-
-
-











Menu Item	Functionality
Purchase Corda Products	Opens the default Web browser to the Purchase Corda Products Web page (www.corda.com/buy).
License	Allows you to enter or view the Corda Builder license key.
About Corda Builder	Provides version information and copyright details for Corda Technologies, Inc. and Corda Builder.

TOOL BAR

The toolbar provides quick access to common Corda Builder features.

Table 2.1 Corda Builder Tool Bar Functions

Button	Functionality
	New Image Template.
	Opens file.
	Saves Image Template.
	Preview in browser.
	Launches the Gallery dialog.
	Launches the Gallery dialog into the Graph category.
	Launches the Gallery dialog into the Gauge category.
	Launches the Gallery dialog into the Shape category.
	Add a Map.
	Add an Image.
	Add a Legend.
	Add a Scale Marker.

Button	Functionality
	Group Objects.
	Delete Object.
	Delete the selected part of graph.
	Undo last action.
	Redo last undo.
	Edit PCScript.
	Edit Sample Code.
	Help.
	Publish.
	Create PDF document with Highwire.

TABS

Located on the bottom-left edge of the Corda Builder UI are two tabs to select the view of the Corda Builder project. Designers that only need the **Design** view can remove the other tabs from the UI by selecting **Edit > Preferences**, selecting the **Publishing** tab, and selecting **Reduce UI for Publishing**.

DESIGN

Use the **Design** tab to view the project graphically and to create or edit the project objects. For more information, see [“Working with Objects” on page 3-2](#).

ITXML

Use the **ITXML** tab to view the project as ITXML code and to edit the ITXML, if desired. For more information about ITXML, see [Appendix C, “ITXML,”](#) of the *Corda 7 Developer Reference*.

- **CORDA BUILDER OVERVIEW**
- *Corda 7 Object Reference*
-
-

CORDA 7 OBJECT REFERENCE

At the core of Corda 7's powerful data representation features is its library of XML properties. The XML object reference provides an overview of each supported XML property, along with any attributes and sub-properties.

To access the Object Reference from Corda Builder, do the following:

- In **Object Properties**, select an attribute or property; then click the Help (?) button at the bottom of **Object Properties** in the Description field. This provides information on the currently selected XML object. For more information on the panes in Corda Builder, see "[Corda Builder Layout](#)" on page 2-6.

USING CORDA BUILDER

This chapter introduces the Corda® Builder™ interface, including the Graph Wizard. Corda Builder gives you complete control over the components of a Corda Builder project. It is also the only interface for creating maps or custom shapes.

This chapter includes the following sections:

- [Working with Objects](#)
- [The Corda 7 Gallery](#)
- [Creating a Basic Graph](#)
- [Creating a Basic Map](#)
- [Troubleshooting Corda Builder](#)

- USING CORDA BUILDER

- Working with Objects

WORKING WITH OBJECTS

Corda Builder projects, and the resulting Image Template files, consist of one or more object components. Supported object types include *graphs*, *maps*, *text boxes*, *legends*, *shapes*, and *imported images*. Manipulate and configure these objects within the project using Corda Builder.

To add an object to a project, select **Edit > Create <object type>**, where *<object type>* is the type of object you want to add.

COMMON OBJECT ACTIONS

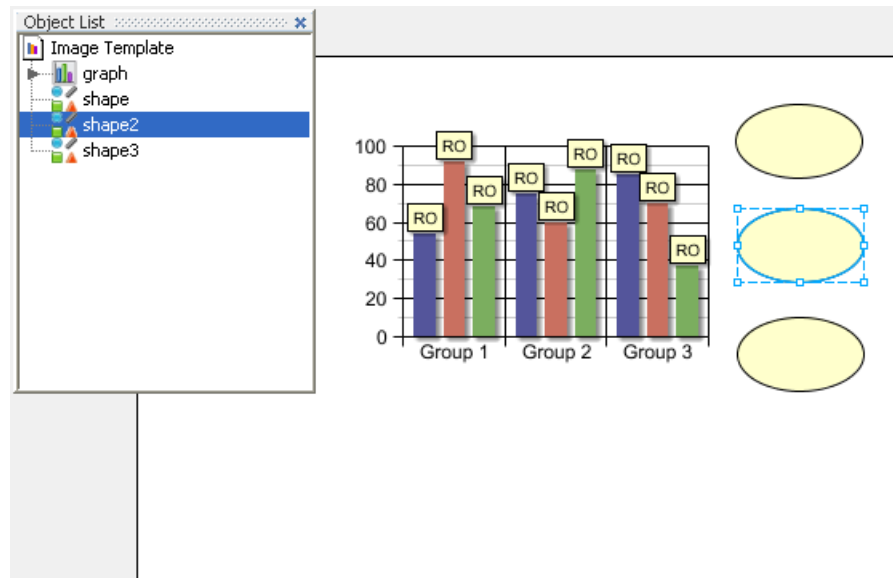
Corda Builder supports several common actions for any object that has been added to a project.

SELECTING OBJECTS

Select an object by clicking it in the [Project Canvas](#), or in the [Object List](#). If an object is currently selected, press the <Tab> key to cycle through the objects in the [Object List](#).

Note: To select a single shape in a map object, or grouped shape object, use the *Select a Point* cursor from *Shape Tools*. For more information, see [“Select a Point” on page 8-13](#).

Corda Builder identifies the selected object by highlighting it in the [Object List](#), and by surrounding the object with a blue, dashed outline in the [Project Canvas](#).



Note: You can also select individual components of a graph or map. To deselect an individual graph or map component, click any white space around the graph or map.

Corda Builder also allows you to select multiple objects by pressing the <Ctrl> key while selecting the desired objects, either on the **Project Canvas** or in the **Object List**. This allows you to move several objects at once.

MOVING OBJECTS

Move a selected object by clicking and dragging it to the desired location with the mouse, or by using the arrow keys on the keyboard to shift the object to the desired location.

RESIZING OBJECTS

Resize a selected object by using the mouse to drag the blue “handles” at the corners and sides of the selected object outline, until the object reaches the desired size, or by holding the <Shift> key and using the arrow keys on the keyboard to resize the object.

MATCHING OBJECT WIDTH OR HEIGHT

To match the width or height of an object to that of another, first select the object that is at the desired width or height; then select the object(s) whose width or height you want to change. From the **Align** view, select **Same Width** (to match the width) or **Same Height** (to match the height).

CHANGING OBJECT PROPERTIES

After an object is selected, access the object’s properties and attributes from **Object Properties**. More information about objects is available in the *Corda 7 Object Reference*. For more information, see “[Corda 7 Object Reference](#)” on page 2-18.

COPYING OBJECTS

To copy a selected object, select **Edit > Copy**, or press <Ctrl>-C.

To paste a copied, or cut, object into the Project Canvas, select **Edit > Paste**, or press <Ctrl>-V.

DELETING OBJECTS

To delete a selected object, select **Edit > Delete**, or press <Delete> or <Backspace>.

Alternatively, cut a selected object, and place it on the system clipboard by selecting **Edit > Cut**, or pressing <Ctrl>-X.

- USING CORDA BUILDER

- Working with Objects












To align objects






ALIGNING OBJECTS

Corda Builder lets you align objects to each other. Aligning objects includes such actions as positioning objects relative to each other, resizing one or more objects relative to another, and moving objects forward or backward in a project's layers.

Note: *Alignment settings such as position and size are not preserved in the project. Once aligned, individual objects can still be sized and moved.*

- 1 Select the object to which other object(s) will be aligned.
- 2 Select the object(s) to be aligned.
- 3 If necessary, select **View > Align** to open the **Align** view.
- 4 From the **Align** view, select the desired alignment option.

Button	Functionality
	Align Left: Aligns the second object, and any subsequently selected objects, with the left edge of the first selected object.
	Align Right: Aligns the second object, and any subsequently selected objects, with the top edge of the first selected object.
	Align Center: Aligns the second object, and any subsequently selected objects, with the horizontal center of the first selected object.
	Align Middle: Aligns the second object, and any subsequently selected objects, with the vertical middle of the first selected object.
	Align Right: Aligns the second object, and any subsequently selected objects, with the right edge of the first selected object.
	Align Bottom: Aligns the second object, and any subsequently selected objects, with the bottom edge of the first selected object.
	Evenly spaces the selected objects horizontally across the Project Canvas.
	Evenly spaces the selected objects vertically across the Project Canvas.
	Resizes the second object, and any subsequently selected objects, to the same width and height as the first selected object.
	Overlay: Resizes the second object, and any subsequently selected objects, to the same width and height as the first selected object, and then overlays the resized objects on top of the first object.
	Resizes the second object, and any subsequently selected objects, to the same width as the first selected object.

Button	Functionality
	Resizes the second object, and any subsequently selected objects, to the same height as the first selected object.
	Moves the selected object to the project's back layer.
	Moves the selected object to the project's front layer.
	Moves the selected object back one layer in the project.
	Moves the selected object forward one layer in the project.

Note: *Overlaying simply changes the height, width, and position of the affected objects. The overlay setting is not preserved in the project—meaning that overlaid objects can still be moved or resized independently, if desired.*

ANCHORING OBJECTS

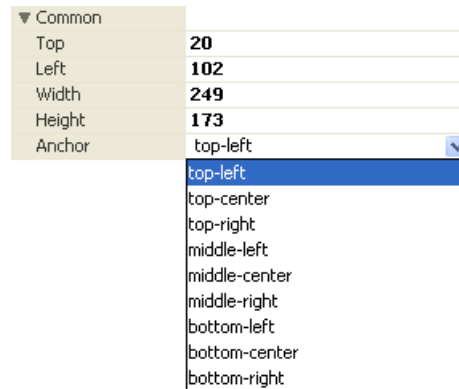
Anchors are used to specify the position of graph, map, legend, text box, and image objects as they expand.

For example, if additional text is added to a text box, the anchor points determine how the text box adjusts for the additional text. To keep the top left corner of the text box stationary so that the text box expands down and to the right as text is added, specify the **Vertical Anchor** as **Top**, and the **Horizontal Anchor** as **Left**.

- USING CORDA BUILDER
- Working with Objects
-
-

To anchor an object

- 1 Select the object to be anchored.
- 2 In **Object Properties**, select the **Common > Anchor** property.
- 3 Select the desired anchor setting.



The anchor setting is a combination of one vertical and one horizontal anchor point. Vertical anchors include **Top**, **Middle**, and **Bottom**. Horizontal anchors include **Left**, **Center**, and **Right**.

RELATIVE POSITIONING AND SIZING

Relative positioning is a special case of object anchor that preserves the location of objects relative to one another. Instead of specifying a fixed position on the **Project Canvas**, use a mathematical expression to specify the object's position relative to another object. Similarly, relative sizing lets you specify how one object grows or shrinks proportionally to another.

To specify relative positioning or sizing, use mathematical expressions in the object's position attributes to describe how the object should be placed in relation another object. The relative position expression always begins with an equals sign (=).

Relative positioning supports use standard arithmetic operators (+, -, *, /) in the relative position expression. For example, to specify that the selected object be positioned 20 pixels past the vertical middle of the object named *Map*, use the following expression:

`=middle(Map) +20`

Similarly, use the division (/) and multiplication (*) operators to specify that the width or height of an object be proportionally linked to the size of another object. For example:

`=width(Map) /2` or `=height(Map) *2`

Because this relationship is saved as part of the project, the objects remain related as defined by the relative position expression.

To position or size an object relative to another

- 1 **Select the object to position.**
- 2 **In *Object Properties*, specify the appropriate relative positioning expressions in the *Common* property.**

The following properties accept relative positioning expressions: *Top*, *Left*, *Width*, *Height*.

- 3 **Press <Tab> or <Enter> to see the results of the positioning expression(s) in the *Project Canvas*.**

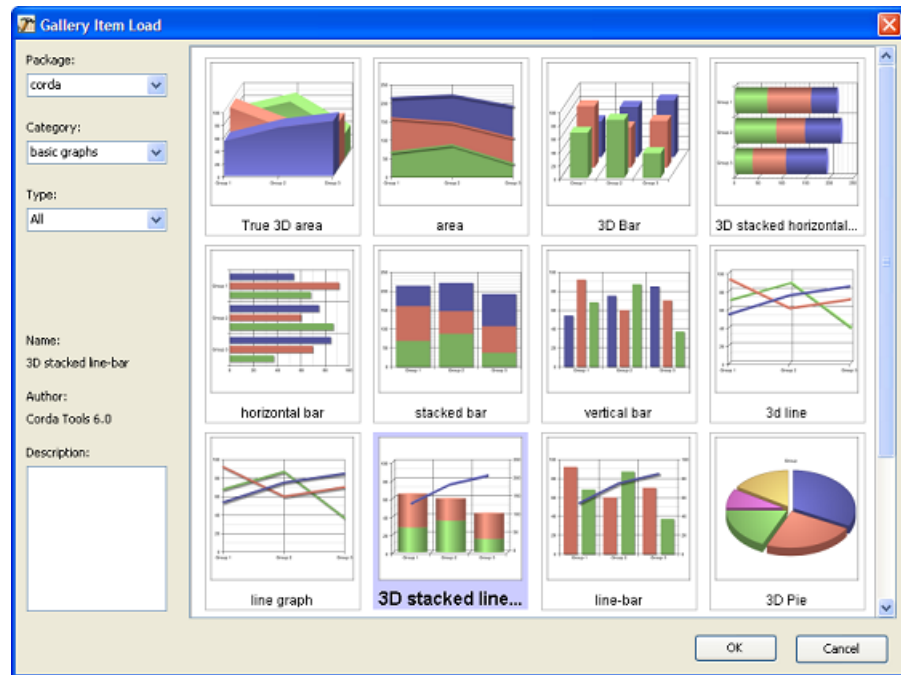
▼ Common	
Top	20
Left	=right(graph)+20
Width	249
Height	173

- USING CORDA BUILDER

- *The Corda 7 Gallery*

THE CORDA 7 GALLERY

The Corda 7 Gallery is a way to store and organize objects that can be used in projects to develop Image Template files. As you use Corda Builder over time to create new objects, Corda 7 Gallery lets you save those objects in a library of project components that you can re-use again and again as circumstances demand.



Additionally, Corda provides a wide variety of sample objects in the Gallery, including shapes, graphs, maps and images. The following sections acquaint you with Corda 7 Gallery:

- [How Gallery Works](#)
- [Saving Objects to the Gallery](#)
- [Loading Objects from the Gallery](#)

HOW GALLERY WORKS

Essentially, Corda 7 Gallery provides easy-to-use access and control of components for creating Image Template files. Corda 7 Gallery provides three levels of organization for component files. Each level of organization corresponds to a folder in the underlying directory structure within which component files are stored.

PACKAGE: Packages are the highest level of organization. For example, a package called *Marketing* might contain all of an organization's Marketing-related object files.

A Package can store object files directly or be further subdivided into object Categories.

CATEGORY: Categories are defined within Packages to further organize object files. For example, within a *Marketing* Package, an organization might create a *Shapes* Category for all shape-based objects used by Marketing.

A Category can store object files directly or be further subdivided into object Types.





TYPE: Types are defined within Categories to further organize object files. For example, within a *Shape* Category, an organization might create a *Polygon* Type for custom shape objects used by Marketing.

Corda Builder stores all Gallery files in <product_root>\Builder\lib\Gallery\. This folder can be further organized into Packages, Categories, and Types as described above.

Note: *The Gallery file storage is separate from the default file location for completed Image Template files that might be accessed by Corda Server. This helps you keep project components and unfinished objects out of your production files.*

LOADING OBJECTS FROM THE GALLERY

Loading an object from the Gallery opens the Gallery interface in Load mode, as shown in the graphic at the start of this topic. Corda Builder's toolbar provides several different Gallery buttons, each opening directly to a different object category in the Gallery:

Button	Functionality
	General Gallery
	Graph Category in Gallery
	Gauge Category in Gallery
	Shape Category in Gallery

- USING CORDA BUILDER
- *The Corda 7 Gallery*
-
-

To load an object from the Gallery

- 1 Click the appropriate **Gallery** button.
- 2 Select the desired object from the **Component Window** and click **OK**.

Regardless of the button you use to open the Gallery, once it is opened you can select different Package, Category, or Type settings to navigate to different areas of the Gallery. Similarly, the Gallery does not restrict how objects are saved in the Gallery. A graph could be saved in the Shape category, or a gauge in the Graph category.

SAVING OBJECTS TO THE GALLERY

Saving a project component to the Gallery makes it readily accessible for use in future Corda Builder projects. The Gallery interface for saving a project component is identical to that used for loading a project component, except for the **New** buttons that appear above the Package, Category, and Type dropdown menus. These buttons let you define new Packages, Categories, and Types as part of the save process.

To save an object to the Gallery

- 1 Select the object to save on the **Project Canvas**.
- 2 Select **File > Save as Gallery Item**.
- 3 (Optional) Create new **Package, Category, and Type** folders to store the project component.

To create a new Package, Category, or Type folder, click the appropriate **New** button, specify a name for the new folder, and click **OK**. The new folder appears in the dropdown list.

- 4 Click **OK** to save the project component to the Gallery.

In addition to letting you save project components, Corda Builder lets you save the **Project Canvas** to the Gallery as well. This lets you create custom project backgrounds and easily apply those settings to other projects as needed.

To save a Project Canvas to the Gallery

- 1 Select the **Project Canvas**.
- 2 Select **File > Save Canvas as Gallery Item**.
- 3 (Optional) Create new **Package, Category, and Type** folders to store the new Gallery item.

To create a new Package, Category, or Type folder, click the appropriate **New** button, specify a name for the new folder, and click **OK**. The new folder appears in the dropdown list.

- 4 Click **OK** to save the current Canvas settings as a component to the Gallery.

Once saved, you can load a saved Canvas just as you would any other project component in the Gallery. When selected, the stored Canvas settings override any existing settings in the **Canvas** property in **Object Properties**.

CREATING A BASIC GRAPH

Once installed, Corda Builder provides robust, intuitive features to help you begin building graph-based Image Template files. Creating a basic graph project involves the following basic steps:

- 1 **Select a Template**
- 2 **Enter Data**
- 3 **Configure Graph Properties**
- 4 **Preview the Project**
- 5 **Save Project**

Note: *These steps are the minimum required to create a simple graph. Creating more complex graph template files can require significant additional effort.*

SELECT A TEMPLATE

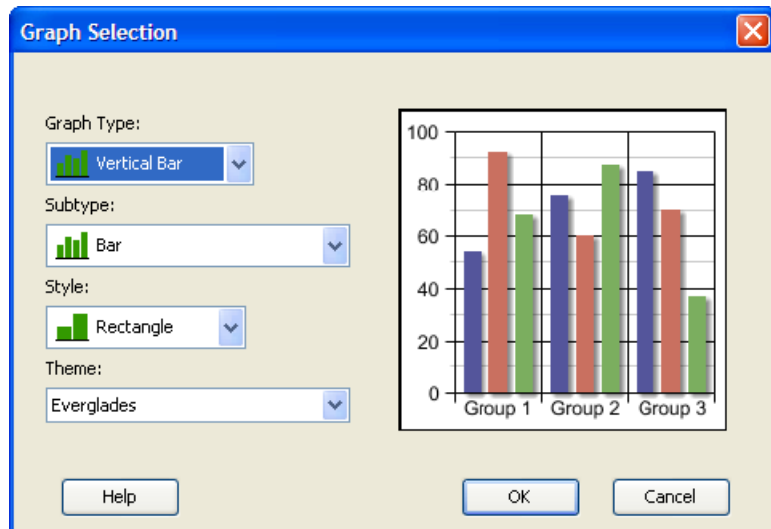
Creating a graph requires that you select a template of the graph property type. Graph templates are available in three possible locations:

USE AN EXISTING IMAGE TEMPLATE FILE You might have an existing graph template already in use that you can leverage to create the new graph. By default, Corda Builder stores Image Template files in <document_root>. Simply open the existing Image Template file and modify as needed.

USE THE GALLERY The Gallery contains many graph components suitable for creating a new graph. Simply open the Gallery and locate the desired graph file.

- USING CORDA BUILDER
- Creating a Basic Graph
-
-

USE THE GRAPH SELECTION TOOL To select a generic graph format to use as a starting point for a graph, select **Insert > Insert Graph**.



The Graph Selection tool has four dropdown menus. Its preview pane displays the currently-specified graph configuration.

Graph Type: Specifies a general graph type, such as Pie or Bar.

Subtype: Specifies a subtype of the selected Graph Type, such as 3D or Stacked.

Style: Specifies a type of look for the graph components, such as Cylinder or Pattern fill. Style isn't available with all Graph Types.

Theme: Specifies a base color theme that Corda Builder applies to the graph.

Note: To learn about the various graph types, see the [Corda 7 Graph Guide](#).

ENTER DATA

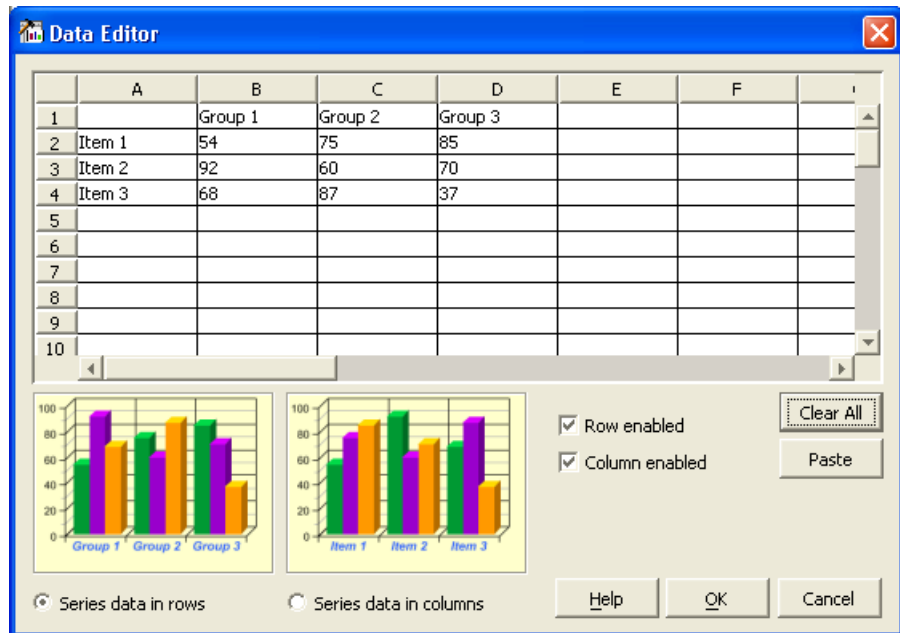
When creating a graph, Corda Builder inserts sample display data. Use Corda Builder's **Data Editor** to edit/replace this data with some that better reflects potential live data for the graph.

To open the data editor, select a graph object and, in **Object Properties**, click the ellipsis [...] button in the **Data Editor** field.

When you first come to the data window, sample data is already in the spreadsheet. Replace this data with your preferred data. (Use the **Clear All** button to erase everything in the spreadsheet.)

As you enter data, the views at the bottom of the dialog change to reflect what you have entered. Two windows appear because, with most standard graphs, the data can be displayed in two ways:

- **Series data in rows.** Each row constitutes a series of data.
- **Series data in columns.** Each column constitutes a series of data.



The preview images are designed to help you decide which setting you want. Check the radio button under the window that correctly displays how the graph orients the data. For more information about data series, see [“Data Interpretation” on page 4-8](#).

In addition to selecting the data orientation, specify the rows and columns of data to enable. For more information on enabling rows and columns, see [“Graphing Specific Columns and Rows” on page 4-11](#).

After entering data and selecting the correct data display option, click **Next** to continue.

Note: *If this project is an Image Template file for Corda Server, this data is typically overridden for the final published image. Enter sample data similar to the expected live data for the published graph to be more confident of the image’s final appearance.*

- USING CORDA BUILDER
- *Creating a Basic Graph*
-
-
-

CONFIGURE GRAPH PROPERTIES

Once the graph is populated with data, configure the look and feel of the graph to suit your needs. Do this by manipulating the graph's XML properties from Object Properties. For more information on graph configuration, see [“General Graph Configuration” on page 4-3](#).

Graph configuration properties include the following:

- General properties: Specifies border, color, etc.
- Title properties: Specifies title text, font, color, etc.
- Graph axis properties: Specifies axis and scale settings
- Grid properties: Specifies the graph's grid settings
- Data label properties: Specifies label display, position, and type
- Hover text properties: Specifies Hover text content, font, border, color, etc.

PREVIEW THE PROJECT



To see how a graph will look in a browser window once it is published, use the Preview button. The **Browser Preview** displays the chart in several different formats by default: Flash*, SVG, and PNG. Change these settings in the **Edit > Preferences** dialog.

For more information, see [“Previewing a Corda Image” on page 13-23](#).

SAVE PROJECT

Once the graph is properly configured, save the project. Depending on your purpose for the graph, save the graph in one of two ways:

SAVE TO <DOCUMENT_ROOT> If you plan on publishing the graph, particularly for use as an Image Template file with Corda Server, save the graph using **File > Save**. Corda Builder saves the file to <document_root> where it is readily accessible by Corda Server.

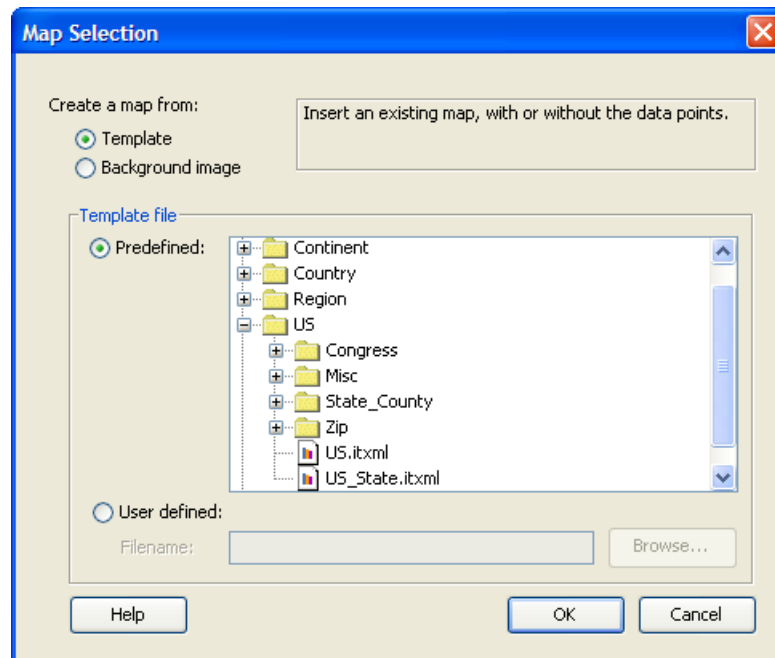
SAVE TO THE GALLERY If the graph is not going to be published, but used or leveraged to build other projects, save the graph to the Corda 7 Gallery by using **File > Save as Gallery Item**. For more information, see [“Saving Objects to the Gallery” on page 3-10](#).

CREATING A BASIC MAP

Once installed, Corda Builder provides robust, intuitive features to help you begin building map-based Image Template files. Since maps are typically pre-defined layered shape structures, you don't have to create the actual map. Rather, select the desired map and customize its XML properties to fit your needs. For more information about maps, see ["Maps" on page 6-1](#).

To create a basic map project

- 1 **Launch Corda Builder.**
- 2 **Click the **Add a Map** button or select **Insert > Insert Map**.**



- 3 **In the Map Selection dialog, select the **Template** radio button; then select the **Predefined** radio button under the **Template file** option.**
- 4 **Browse through and select the **US.itxml** template from the **US** folder, and click **OK**.**

A map of the United States appears in Corda Builder.

This template allows you to map data for the United States on a state-by-state basis. There are many other map templates to choose from. For more information about available map templates, see the *Corda 7 Map Guide*.

- USING CORDA BUILDER

- Creating a Basic Map

5 From the **Object List**, select the **US > States > CA** object.

6 In **Object Properties**, open the **Shape Settings** property and change the **Value** attribute from *no-data* to *100.0*. Press <Tab> or <Enter> to accept the value change.

Note that the state of *California* changes colors to reflect the new data value you just entered for *California*.

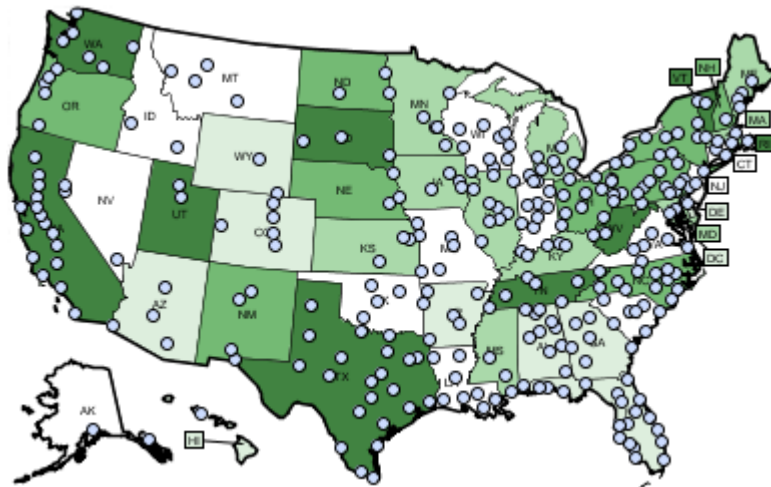
Note: *The US map in this example consists of 50 data items; each is a shape that corresponds to one of the states. Assign data to any data item in a map. This map also contains city data items—the blue dots on the map. The cities do not appear in the final published version of the map because we never assign data to these data items.*

7 From the **Object List**, select the **States** object. In **Object Properties**, click the **Ellipsis** button [...] to open the **Data Editor**.

This is another way of editing data in a map. It lets you edit values for all data items in the selected map layer at once.

8 Enter in values for several states, and click **OK**.

Notice that the states whose values you changed are now colored differently. The color of each state reflects the range of the value that you assigned to it.



Important: *When creating a map to use as a template with Corda Server, the data in **Data Editor** is irrelevant, but it helps you visualize what the map looks like when populated with data.*

9 Preview the map by clicking the **Preview** button or by selecting **File > Preview**.

This lets you see how the map looks and behaves in the Web environment. For more information about Previewing Image Template files, see [“Previewing a Corda Image” on page 13-23](#).

- 10 **Save the map as an Image Template file by clicking the **Save Project** button or by selecting **File > Save**.**

Save the map as Map1.itxml.

Saving a project file allows you to return to this project later and make modifications. More importantly, Corda Server can use the project file as an Image Template file (template) for dynamic graphs.

By default, Corda Server retrieves Image Template files from <document_root>.

To publish this map as a Web page or an image, select **File > Publish**. For more information about the publishing process, see [“Publishing a Project” on page 20-2](#).



USING CORDA BUILDER

Troubleshooting Corda Builder

TROUBLESHOOTING CORDA BUILDER

This section provides common troubleshooting information and answers frequently asked questions for Corda® Builder.

Topics in this section include the following:

- [Updating Corda Builder](#)
- [Known Issues](#)
- [Installation Problems](#)
- [Setup Problems](#)
- [Image Generation Problems](#)

To view the latest troubleshooting information, visit the following Web pages:

- Troubleshooting: <http://www.corda.com/support>
- Frequently Asked Questions: <http://www.corda.com/support>

To contact technical support, email support@corda.com.

Note: *Users of freeware versions of Corda Builder and Corda 7 are not entitled to Corda technical support.*

UPDATING CORDA BUILDER

Corda Technologies provides frequent updates to Corda Builder. These updates both enhance existing features and resolve issues in the previous versions. Oftentimes, the problem can be solved simply by updating to the latest version of Corda Builder.

Note: *Corda recommends updating Corda Builder to resolve problems or to add new features in which you are interested. This helps minimize the risk of unforeseen consequences related to product updates.*

To automatically check for updates, select **Help>Check for Updates** with Corda Builder. Visit the Corda Technologies Web site (<http://www.corda.com>) regularly to learn about the latest patches and updates.

KNOWN ISSUES

The following is a list of known issues with Corda Builder on various platforms.

GENERAL

- You get poor results when you overlap imported images with graph objects when generating a PDF or EPS image.
- In some instances, the Corda 7 installer does not run with a display resolution of 16 colors. Set the resolution to 256 colors or higher.

WINDOWS

There are no known issues at this time.

MAC OS X

- When you double-click an Image Template file (.itxml) with a Mac* OS X, it launches Corda Builder, but Corda Builder does not open the file. It does, however, open the file if you double-click it again.

SOLARIS

There are no known issues with Solaris* at this time.

LINUX

There are no known issues with Linux* at this time.

INSTALLATION PROBLEMS

This section contains troubleshooting tips for problems that occur during the installation of Corda Builder.

INSTALLATION PROGRAM DOES NOT RUN

In some instances, the Corda 7 installer does not run with a display resolution of 16 colors. Set the resolution to 256 colors or higher.

SETUP PROBLEMS

This section contains troubleshooting tips for problems that occur during the setup of Corda Builder.

None at this time.

- USING CORDA BUILDER
- *Troubleshooting Corda Builder*
-
-

IMAGE GENERATION PROBLEMS

This section contains troubleshooting tips for problems that occur with image generation.

CORDA SERVER POSITIONS MY OBJECTS DIFFERENTLY THAN HOW I POSITIONED THEM IN CORDA BUILDER

Data that you send to Corda Server can differ dramatically from the sample data in an Image Template file. Because of this, Corda Server may need to automatically resize objects and the Image Template file to prevent overlap. If you want to make sure that an object stays anchored to the same place that you put it in Corda Builder, use the Corda Builder anchoring feature.

If desired, disable automatic resizing in the Image Template file through Corda Builder.

TROUBLE DISPLAYING INTERNATIONAL CHARACTERS IN HELVETICA, TIMES, OR COURIER FONTS

Corda Builder supports some international characters when using the pre-installed Lucida font set, but not when using the Times, Helvetica, and Courier font sets. However, Asian characters are not supported with any of the pre-installed font sets. If you need international character support for Asian character sets, import a custom font.

For information about custom fonts, see [“Corda Fonts” on page 11-32](#).

CERTAIN CHARACTERS SHOW UP AS A BOX IN CORDA BUILDER

Some fonts are unable to display certain characters. Helvetica, for example, is limited in the number of characters beyond 255. Change the font that is selected for that text to another font, such as Lucida Sans. With international characters, you may need to create a custom font for Corda Builder using the Corda Font Converter. This also helps with double-byte characters.

For information about custom fonts, see [“Corda Fonts” on page 11-32](#).

CHARACTERS NOT DISPLAYING PROPERLY IN STRINGS PASSED TO JAVASCRIPT FUNCTION THROUGH DRILLDOWN

By default, Corda Builder URL-encodes all drilldown effects. In other words, drilldown strings are encoded in such a manner that browsers are able to properly interpret the intended drilldown destination.

Unfortunately, this creates a problem when the intended target of a drilldown effect is a Javascript function instead of a URL. Corda Builder may URL-encode characters that should be Javascript-encoded instead. Because Javascript functions do not recognize URL encoding, the characters display improperly.

This situation is further complicated by occasions when strings in Javascript function should be URL-encoded because the string is eventually interpreted by a browser as a URL.

Because this is often the case, Corda Builder cannot simply attempt to detect whether the target is a URL or a Javascript function. Therefore, it just assumes that it should encode it as a URL.

Anytime a drilldown effect passes a string containing a character higher than 127 (0x7E) through a Javascript function and this string is not supposed to be interpreted as a URL, the string is not interpreted properly. To get around this behavior, take advantage of the `%_JS_ENCODE` meta tag. This meta tag instructs Corda Builder to Javascript-encode the drilldown effect rather than URL-encode it. It should be included at the end of any drilldown effect that needs to be Javascript encoded.

For example, the following drilldown effect does not display correctly. Instead of *Group*™, you see *Group %e2%84%a2*.

```
Javascript:alert("Category: Group ™")
```

However, adding the `%_JS_ENCODE` meta tag to the end of the drilldown effect corrects the problem.

```
Javascript:alert("Category: Group ™")%_JS_ENCODE
```

Note that you would not want to use this meta tag if your string was to be interpreted as a URL, as in the string below because *Group*™ needs to be interpreted as *Group %e2%84%a2* for the browser to properly recognize it.

```
Javascript>window.open("http://test.com/?caterwaul ™")
```

DRILLDOWN, HOVER, OR NOTES NOT APPEARING IN CORRECT PLACE FOR X-Y, TIME, OR PARETO GRAPHS

In graphs where data is sorted (X-Y, time, or Pareto graphs where *Properties > Graph Properties > General > Sort Data* has been checked), the target “category” for hover text, drilldown, and notes is enumerated by sort order rather than list order. Because of this, you may experience unexpected behavior when using these features with the specified graph types.

At present, there is no workaround for this problem when producing static graphs in Corda Builder. Uncheck *Sort Data* to disable sorting in X-Y and Time graphs.

DRILLDOWN, HOVER, OR ROLLOVER DATA LABEL PROBLEMS WITH PNG OR JPEG IMAGES

These features are not available in PNG or JPEG images without using the Corda Embedder.

Certain browsers, including browsers older than Netscape* 4, Internet Explorer* 4, or Opera* 7, are unable to display any interactive features such as hover text and drilldown effects in PNG or JPEG images. Some browsers, including Netscape 4.X, provide only limited interactivity via image maps (see last paragraph).

- USING CORDA BUILDER
- Troubleshooting Corda Builder
-
-

By default, these features are supported in PNG and JPEG images via Javascript. If Javascript is disabled for a browser, these features are instead supported by image maps. If you have disabled **JavaScript Hover** in the **User Preferences > Publishing** dialog, these features are also supported by image maps.

When these features are supported by image maps, feedback (i.e., response time between placing a mouse over a data item and seeing hover or rollover text) is slow. You also cannot customize the appearance of rollover data labels and hover text. In **Netscape** browsers, you may be unable to get hover text without also enabling a drilldown effect for the specified data item.

NO DRILLDOWN, HOVER, OR ROLLOVER DATA LABELS IN PDF, EPS, AND TIFF IMAGES

These features are not supported in PDF, EPS, and TIFF images.

GRAPH OR MAP ANIMATIONS SLOW AND CHOPPY WITH AUTO-UPDATING TURNED ON

The automatically updated image feature in Flash is not compatible with animations.

AUTO-UPDATING FLASH IMAGES DO NOT REFRESH CONTENT CORRECTLY

On some browsers you may experience caching problems with auto-updating Flash images. To get around these problems, set the **Content Expiration** header for each Flash image you load to expire immediately. See [“Solving Caching Issues with Auto-Updating Flash images” on page 12-5](#).

GRAPHS

This chapter provides an introduction to general graph concepts and graph types. The *Corda 7 Graph Guide* provides more detailed information about each graph type, including the type of data each graph type accepts as well as what the graph can be used for; the graph sub-types supported by each graph, and how any special features that might be available can be implemented.

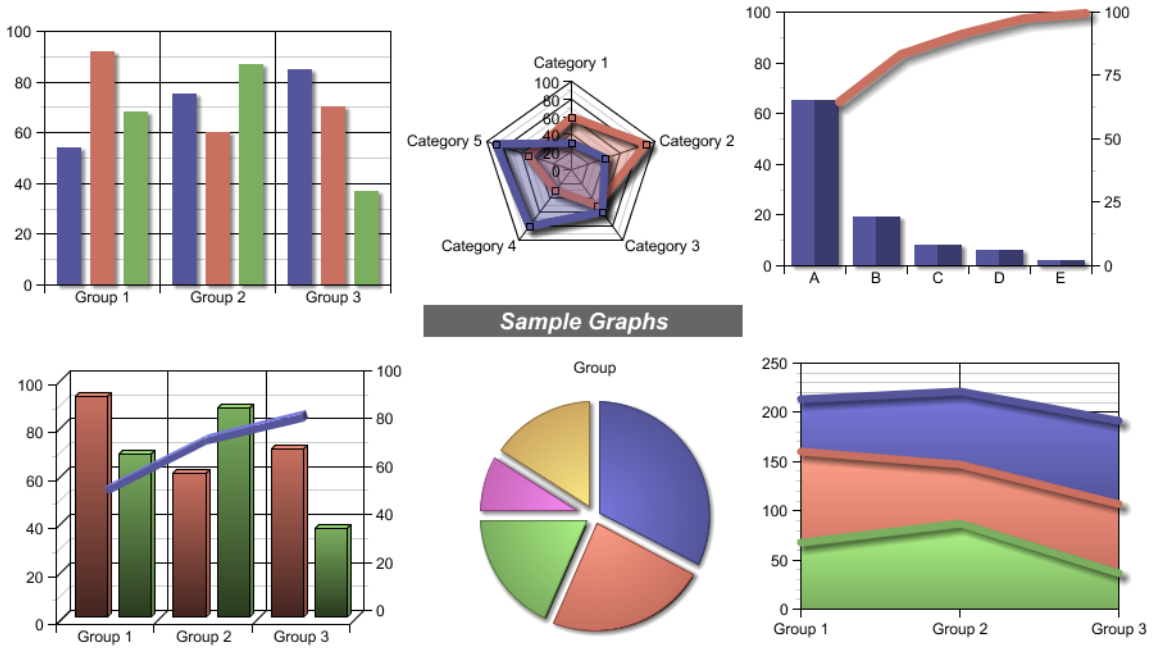
This chapter includes information on the following graph-related topics:

- [General Graph Configuration](#)
- [Graph Data](#)
- [Bar-Based Graphs](#)
- [Pie Graphs](#)
- [Line-Based and Plot Graphs](#)
- [Heat Maps](#)
- [Gauges](#)

Note: *When creating a graph, you might also want to create text boxes for the graph title and a legend. These objects are discussed in Chapter 8, “[Other Corda Objects](#).”*

GRAPHS

Add, move, and delete graphs within a project as you would any other object in a Corda Builder project. For more information, see [“Working with Objects”](#) on page 3-2. For information about creating a basic graph, see [“Creating a Basic Graph”](#) on page 3-12.



GENERAL GRAPH CONFIGURATION

Several configuration options are common to all graphs. Most of these involve setting specific graph properties, which are described in the *Corda 7 Object Reference*. However, the following sections provide additional detail.

GRAPH BOUNDARIES

When you create a graph, all parts of the graph are placed within predefined boundaries. This includes everything on the grid area, as well as text, tick marks, and scales.

However, as the scales and scale labels grow (as a result of changing data from a template or when sent to Corda Server, for example), the grid area might shrink to compensate. To avoid this, set the tick marks and scales outside of these boundaries so the grid area always remains the same size.

This is particularly useful if graph label text is larger than the graph itself. If labels extend beyond the project border, the label field automatically grows to fit the label unless the label width is fixed. This option is also beneficial when two graphs are combined (overlaid) for data comparisons because it is easy to align the grid areas.

Properties and attributes related to graph boundaries are available from the `Graph Settings` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

GRAPH SERIES COLOR

Each series of data is typically represented by a different color. Series colors can be set individually or as a group by changing the color theme.

Properties and attributes related to graph series color are available from the `Bar Series` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

GRADIENT FILL

`Bar`, `Area`, and `Pie` graphs support gradient fill for their graph components.

Properties and attributes related to gradient fill are available from the `Bar Settings` (bar graphs), `Pie Settings` (pie graphs), and `Area Settings` (area graphs) in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- **GRAPHS**

- *General Graph Configuration*
-
-

PATTERN FILL

Typically, each series of data in a graph is represented by a different color. However, it might be desirable (especially for black and white printing or for color-blind viewers) to use patterns in addition to colors.

Patterns can be used in **Bar**, **Area**, and **Pie** graphs (**Line**-based graphs support black and white printing through line styles). See [“Line Width, Color, and Style” on page 4-20](#).

Conceptually speaking, patterns work exactly like color themes. Each series uses a different pattern style. Choose a pattern theme to dictate which series uses which style.

Patterns use the colors that have been specified in the color theme (see the previous section). By default, the pattern is placed against a white background, but the background color can be changed. This background color applies to all of the patterns. The foreground colors are controlled by the color theme on the left side of this dialog.

Corda Builder automatically selects a different pattern for each series on the map, using the color associated with that series.

While Corda Builder does not let you add new pattern styles, it does allow you to change the pattern styles assigned to each data series.

There are sixteen different pattern styles. When the number of series in a graph surpasses sixteen, Corda Builder wraps around and starts using the first pattern again.

Properties and attributes related to pattern fill are available from the **Bar Settings** (bar graphs), **Pie Settings** (pie graphs), and **Area Settings** (area graphs) in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

SHADOW

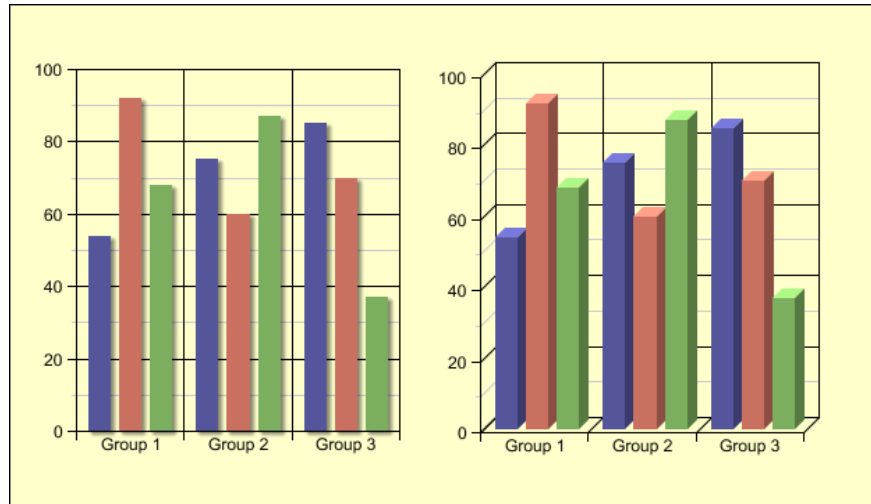
Shadow effects help graph components stand out. Shadow effects are fully configurable, including color, width, and type. Shadow styles include *Fade*, *Glow*, and *Hard*.

Note: *Shadows are not used with 3D graphs (see the next section).*

Properties and attributes related to shadow effects are available from the **Shadow Settings** property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

3D

3D effects are a way to enhance the display of information in graphs. Many graph types allow you to use 3D effects.



Some 3D graph types include the ability to configure the 3D environment.

Properties and attributes related to 3D graph settings are available from the [3D Settings](#) property from [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- **GRAPHS**
- *Graph Data*
-
-

GRAPH DATA

This section discusses how to manipulate static graph data in Corda Builder. If the graph is used as part of an Image Template file with Corda Server™, the static data is used for sample or design purposes only. Dynamic data is supplied when the Image Template file is published with Corda Server.

This section includes the following topics:

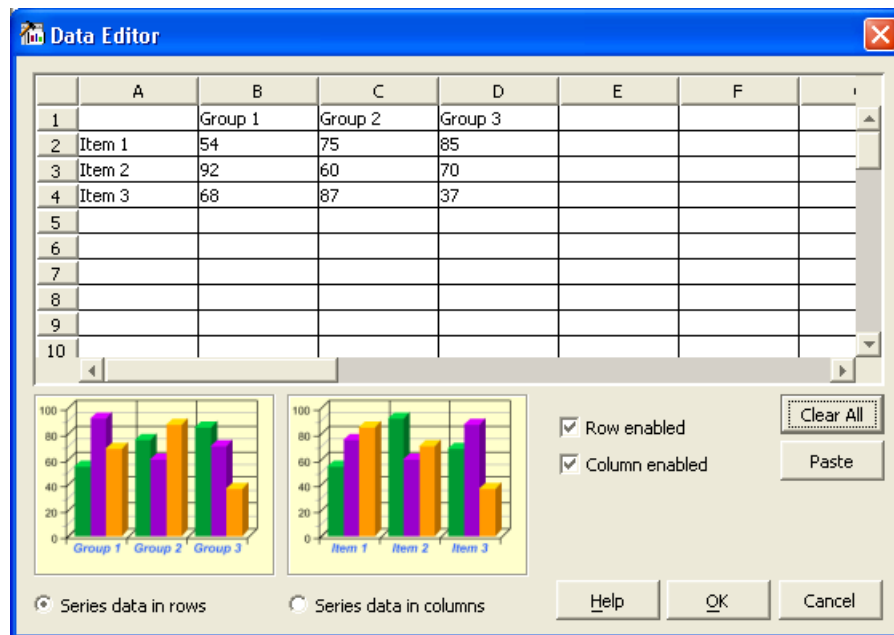
- [Data Input](#)
- [Data Interpretation](#)
- [Column and Row Names](#)
- [Graphing Specific Columns and Rows](#)
- [Transposing Columns and Rows](#)

DATA INPUT

When creating a graph, Corda Builder inserts sample display data. Use Corda Builder's [Data Editor](#) to edit/replace this data with some that better reflects potential live data for the graph.

To open the data editor, select a graph, and then select [Properties > Data Editor](#); or right-click a graph and select [Data Editor](#).

The **Data Editor** lets you manipulate graph data in a spreadsheet-like interface; you can add, edit, and identify what data to include in the graph. Add data to the **Data Editor** by copying data from an existing data source or by entering the data manually.



COPYING DATA FROM AN EXISTING DATA SOURCE

To create graphs using data from an existing data file, such as a spreadsheet or word processing application, copy that data to the clipboard and paste it into the **Data Editor**. To use data from an HTML table in a Web page, first copy the contents of the table to a spreadsheet and then copy the contents from the spreadsheet to **Corda Builder**.

ENTERING DATA MANUALLY

If data is not contained in a spreadsheet or table, create the graph by entering the data manually.

- GRAPHS
- *Graph Data*
-
-

To manually enter data

- 1 **Open the *Data Editor*.**
- 2 **Select the cell where you want to enter the data.**
- 3 **Enter the first value and press <Tab> or <Enter>.**

Press <Tab> to move to the next column, press <Enter> to move to the next row, or use the arrow keys to navigate.

As you move to the next cell, the graph preview at the bottom of the *Data Editor* changes to reflect the new data.

- 4 **When you are done entering data, click **OK** to save the settings.**
- 5 **Clear data from one or more *Data Editor* cells by selecting the cells and pressing the <Delete> key.**
- 6 **Clear all data from the *Data Editor* by selecting a single cell and pressing **Clear All**.**

DATA INTERPRETATION

The building block of every graph is a *data item*. Data items vary depending on the graph. For example, a data item in a bar graph is a single data value represented as a bar. A data item in an X-Y bubble graph, however, consists of three data values—an X-value, a Y-value, and a Bubble value. Grouped together, all of the data items in a graph can be referred to as a *data set*.

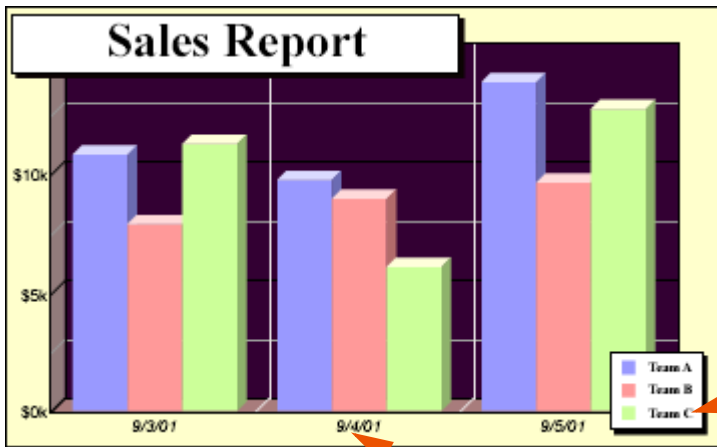
Corda Builder groups data sets into *series* and *categories*. Typically, *series* correspond to rows and *categories* to columns on a spreadsheet, although some graphs may transpose the rows and columns.

All data items in a specific data *series* are displayed in the same color. In line graphs, they are also connected by a line. The series is generally the data that you want to emphasize and the series title appears in the legend. Each item in a series of data belongs to a different *category*. All items in the same category are grouped together in the graph; the category

titles appear in the category scale. [Example 4.1](#) illustrates the roles of series and categories of data in creating a graph.

Example 4.1

Series and Categories



	Categories		
Series	9/3/01	9/4/01	9/5/01
Team A	10789	9745	13841
Team B	7861	8916	9612
Team C	11263	6047	12743

Each value on this row is a data item in the **Team C series** of data. The data items appear as green bars.

Each value in this column is considered to be a data item in the **9/4/01 category**. They are grouped together in the graph.

For all variations of Bar, Line, and Radar graphs, insert data in the **Data Editor** as shown in [Example 4.1](#). Pie graph data is similar, but only has one category. Pareto graph data only has one series. The data in more complex graphs, such as box plot, stock, X-Y, and time plot graphs, is organized somewhat differently, although the basic concept of series and categories remains the same. To learn exactly how to organize data for these graph types, see Chapter 2, “[Data Organization](#),” in the *Corda 7 Graph Guide*.

CATEGORIES AND SERIES: AN EXAMPLE

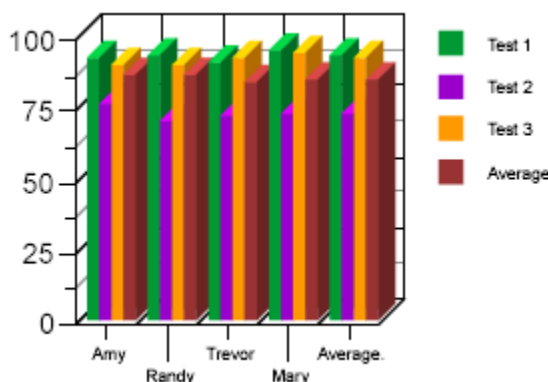
Consider the following example of gathering statistics on a specific college exam: There are three versions of the exam (Test 1, Test 2, and Test 3), along with data about the results of each exam for four students: Amy, Randy, Trevor, and Mary. The graph should represent all

- GRAPHS
- Graph Data
-
-

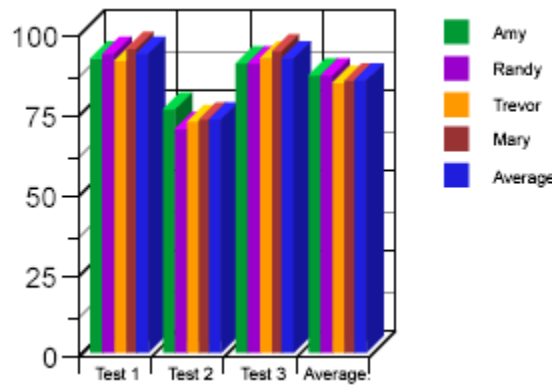
this information, and show average test scores for each test and for each student. The graph data is extracted from the following spreadsheet:

	A	B	C	D	E	F
1		Amy	Randy	Trevor	Mary	Average Test Score
2	Test 1	92	93	91	95	93
3	Test 2	76	70	72	73	73
4	Test 3	90	90	92	94	92
5	Average Testers Results	86	86	84	85	85
6						
7						
8						
9						

Depending on the graph's purpose, either student scores for all tests, or average student score for a specific test can be emphasized in the graph by defining it as the series data. Since the source spreadsheet has test scores for each student in rows, Corda Builder uses these as series data by default. To change the focus of the graph to a student's scores across all three tests, select **Series data in columns**. The resulting graphs are quite different.



Series data in rows



Series data in columns

COLUMN AND ROW NAMES

When importing data from an external application, the first column and row are assumed to be headings, and are used to create names for each category and series of data. The series name is used in the legend; the category name is used along the category scale. These values can be modified in the **Data Editor** as needed.

Note: *Some graph types, such as X-Y or time plot, do not use categorical data. For these graphs, the concept of category names is not applicable, and the first row of the spreadsheet contains regular data.*

GRAPHING SPECIFIC COLUMNS AND ROWS

Sometimes it is desirable to prevent certain data rows or columns from being graphed. Although removing those rows or columns solves the problem, this may not be an acceptable solution if the data set might be used in other situations. Instead, use the **Data Editor** to hide the rows and columns that should not be graphed by selecting the row or column and unchecking the **Row Enable** or **Column Enable** checkbox.

Use Row Enable and Column Enable to include or exclude data from the graph.

	A	B	C	D	E	F
1		Amy	Randy	Trevor	Mary	Average Te...
2	Test 1	92	93	91	95	93
3	Test 2	76	70	72	73	73
4	Test 3	90	90	92	94	92
5	Average Te...	86	86	84	85	85
6						
7						
8						
9						
10						

Row enabled
 Column enabled

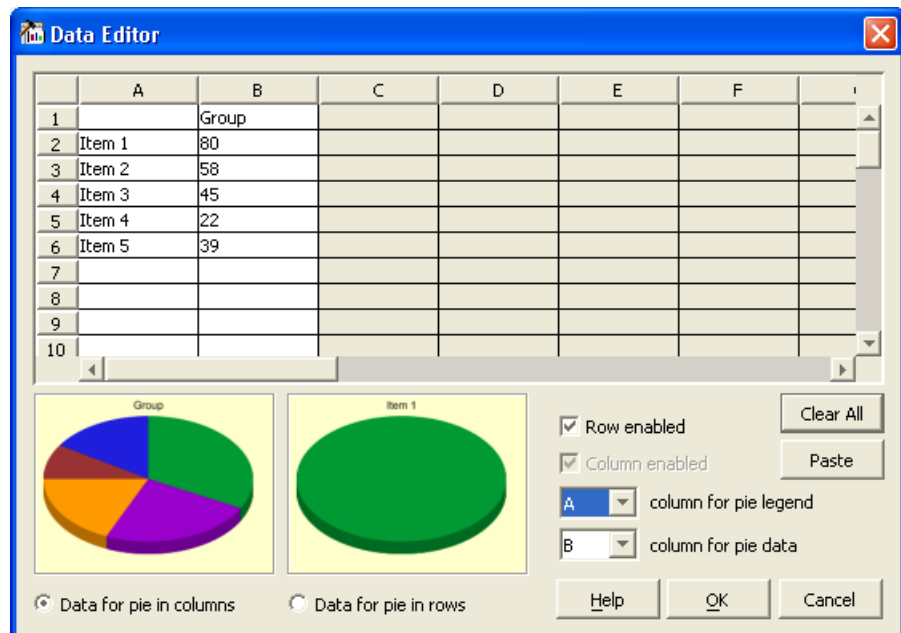
Series data in rows
 Series data in columns

- GRAPHS
- Graph Data
-
-

PIE GRAPH DATA

Because pie graphs have only one category of data, use the **Data Editor** to specify the data column used for the pie data. There are two settings in the **Data Editor** to accomplish this:

- **Column / Row for pie legend.** Specifies which column (or row) of data to use for the pie graph legend. The pie legend and data can be the same column.
- **Column / Row for pie data.** Specifies which column (or row) of data to use for the pie graph.



TRANSPOSING COLUMNS AND ROWS

Transposing rows and columns effectively interchanges the series and category data in the graph. The Data Editor defaults to **Series Data in Rows**, which specifies that categories are represented by columns, and series are represented by rows. To transpose the graph data, select **Series Data in Columns** to specify that categories are represented by rows and series are represented by columns.

BAR-BASED GRAPHS

The following basic formatting options are available for bar-based graphs, which include [Vertical Bar](#), [Horizontal Bar](#), [Line Bar Combo](#), and [Pareto](#) graphs:

- [Bar Width and Spacing](#)
- [Bar Borders](#)
- [Floating Stacked Bars](#)
- [Bar Images](#)
- [Bar Graph Data Ranges](#)

BAR WIDTH AND SPACING

All bar graphs give you the option of changing the width of the bars within the graph. Except for Pareto graphs, bar graphs also allow you to specify bar spacing.

BAR WIDTH

This value can be somewhat misleading in situations where you have more than one bar in a category. It represents the percentage of category space reserved for displaying the bar(s) in that category. Any remaining space is white space, so bar width effectively determines the white space in a category.

By default, **Corda Builder** determines bar width based on graph format settings and the available display area. For example, if the width available to display each category is 1 inch, and the bar width is set to 80 percent, then the width of all bars in the category is 8/10 of an inch, or 80 percent of the category width.

However, **Corda Builder** supports forcing all bars in a graph to the same width. This can be useful in some situations but can cause bars to overlap since a fixed bar width cannot compensate for changing graph data.

Properties and attributes related to bar width are available from the [Bar Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

BAR SPACING

When a category contains several bars, **Corda Builder** lets you customize the white space between each bar in that category.

The space between bars is a percentage of the bar width. For example, if the space between bars is set to 15 percent, then the space between each bar is 15 percent of the width of the bar.

- **GRAPHS**

- *Bar-Based Graphs*

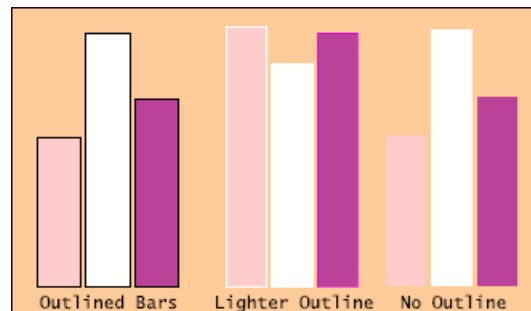
Bar spacing does not affect the way that bar width is calculated. The width from the start of the first bar to the end of the last bar in a category is always the same—the value specified by bar width—no matter how large the bar spacing value.

This number has no effect when there is only one bar in each category.

Properties and attributes related to bar spacing are available from the `Bar Settings` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

BAR BORDERS

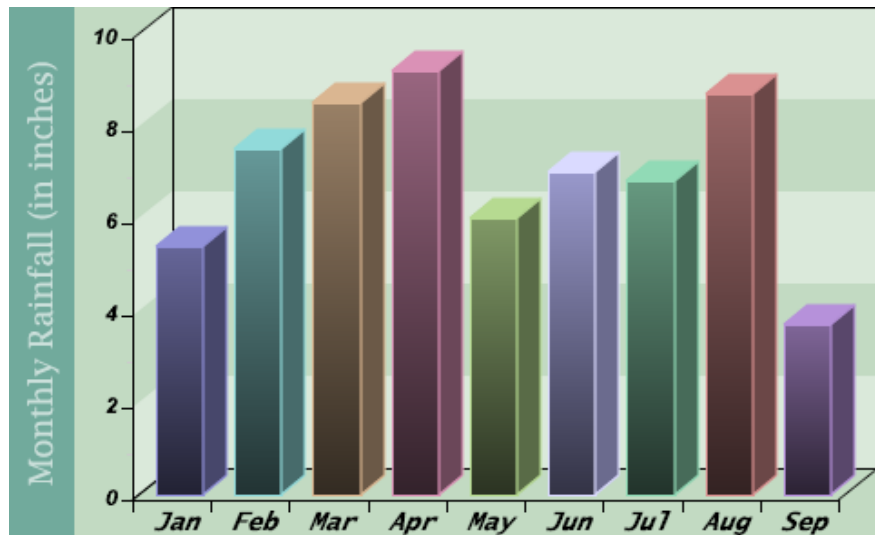
Bar borders configured in Corda Builder are applied to all bars in the graph and cannot be set on a bar-by-bar basis. Use PCScript or ITXML to set a border on a bar-by-bar basis. For more information, see the *Corda 7 Graph Guide*.



Properties and attributes related to bar borders are available from the `Bar Settings` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

MULTICOLOR MODE

By default, if a bar graph has only one series of data, all of the bars are the same color. However, a graph is often more visually appealing if it features more than one color. The graph theme determines the colors used (see “[Graph Series Color](#)” on page 4-3).



Properties and attributes related to multicolor series are available from the [Bar Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

FLOATING STACKED BARS

Both vertical and horizontal stacked bar graphs can be configured as floating stacked bar graphs. In a floating stacked bar graph, the bottom segment (the last data item) in each stacked bar is invisible. This has the effect of making the bar seem to float.

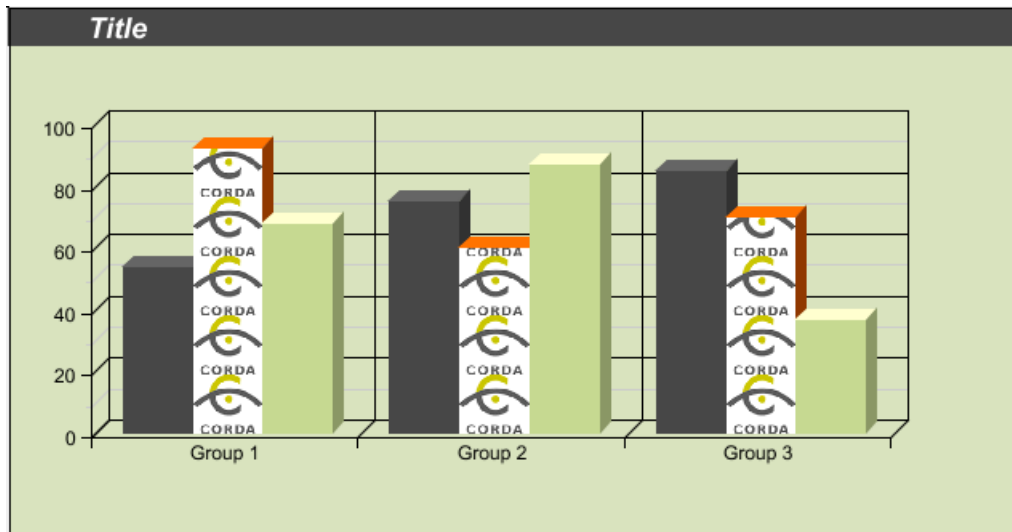
Properties and attributes related to floating stacked bars are available from the [Bar Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- GRAPHS
- *Bar-Based Graphs*
-
-

BAR IMAGES

Corda Builder supports the insertion of images on bar faces in a graph. By default, the image is tiled vertically, but it can be stretched across a bar, if desired. The selected bar image also identifies the series in the graph legend.


The default location for graph images is `<product_root>\Resources\image_templates`.



Properties and attributes related to bar images are available from the [Bar Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

BAR GRAPH DATA RANGES

Bar graphs provide limited data range capabilities. These range capabilities are similar to map ranges in that they override the default series properties. Bar graph ranges let you control data display without having to modify the properties of each data item. Bar graphs support the following range-related features:

ADD A RANGE: Click the **Add** button  in the **Ranges** property. To add multiple ranges simultaneously, click the ellipsis [...] in the **Generate** field to open the **Generate Graph Ranges** dialog.

DELETE A RANGE: Click the **Remove** button  in the appropriate **Range** property.

MODIFY A RANGE: Modify the desired properties in the appropriate **Range** property.

Properties and attributes related to bar graph data ranges are available from the **Ranges** property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- GRAPHS
- Pie Graphs
-
-

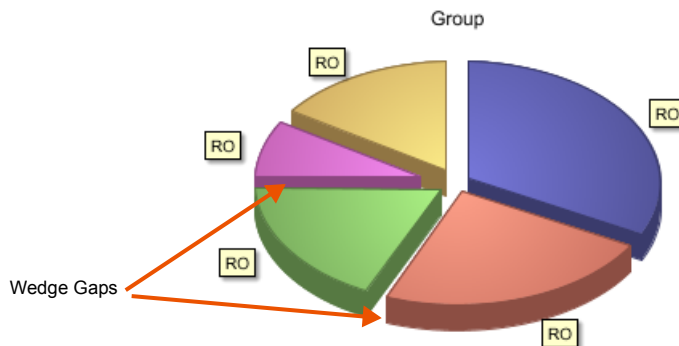
PIE GRAPHS

The following basic formatting options are available for pie graphs:

- [Pie Wedge Gaps](#)
- [Exploded Pie Wedge](#)
- [Pie Wedge Outlines](#)

PIE WEDGE GAPS

Accent pie wedges in both 2D and 3D pie graphs by putting gaps between the pie wedges.



Properties and attributes related to pie wedge gaps are available from the [Pie Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

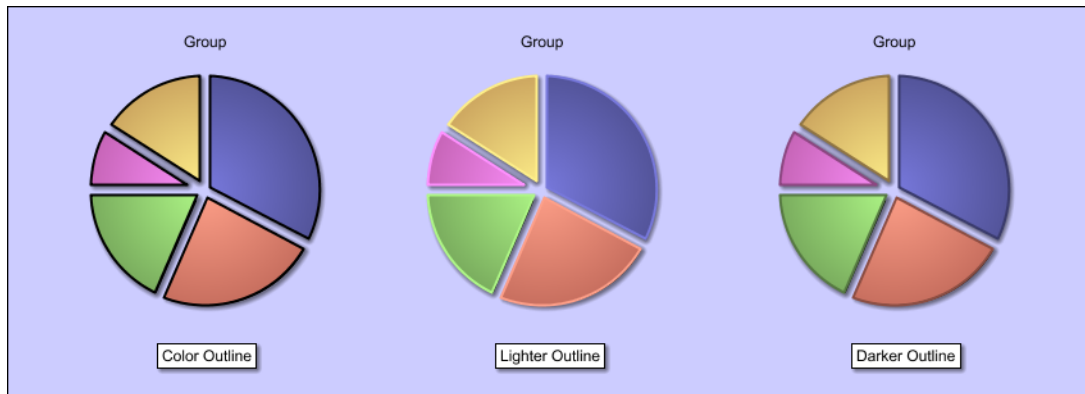
EXPLODED PIE WEDGE

This option allows you to select a single data series wedge to accent at a greater distance from the pie.

Properties and attributes related to exploded pie wedges are available from the [Pie Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

PIE WEDGE OUTLINES

Pie wedge outlines configured in Corda Builder are applied to all wedges in the graph and cannot be set on a wedge-by-wedge basis.



Properties and attributes related to pie wedge outlines are available from the [Pie Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- **GRAPHS**

- *Line-Based and Plot Graphs*

LINE-BASED AND PLOT GRAPHS

The following basic formatting options are available for line-based and plot graphs, which include [Line](#), [Line Bar Combo](#), [X-Y Plot](#), [Time Plot](#), [Radar](#), and [Pareto Graphs](#). These graphs share many common characteristics. For example, each data item is represented as a distinct, plotted point, at which a symbol can be placed; and the data items in each series can be connected by a line.

- [Line Width, Color, and Style](#)
- [Series Symbols or Images](#)
- [Bubble Color and Size](#)
- [Radar Graph Transparency](#)

LINE WIDTH, COLOR, AND STYLE

With the exception of the [Scatter](#) subtypes, [Line](#), [Line Bar](#), [X-Y](#), [Time Plot](#), [Pareto](#), and [Radar](#) graphs all support lines for graphing data series'. The line sequentially connects each data point in the series. Lines can be customized on a series-by-series basis, and some graphs support coloring the area below a line.

A line can be displayed using a number of different line styles. By default, each line is solid. However, alternate styles are available, including dashed lines, dotted lines, and dashed-dotted lines.

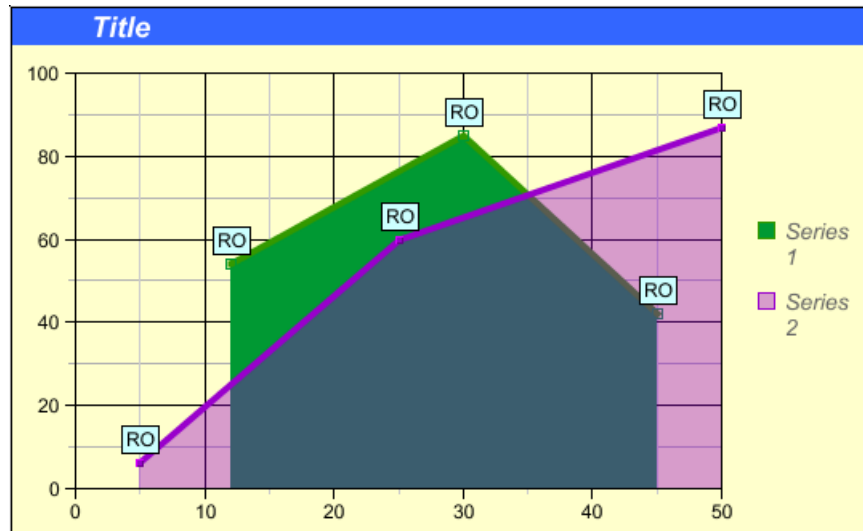
Note: *The style of a 3D line cannot be changed.*

Changing the width or color of any line does not change the color of symbols, filled areas, or bubbles associated with the series.

Overriding line color replaces the default theme color with the specified line color.

[X-Y](#), [Time Plot](#), and [Radar](#) graphs support coloring the region beneath a line. The fill does not have to match the line color.

Note: You do not have to enable a line to color in the area beneath a line.



Properties and attributes related to line style, color, and width are available from the `Line Series` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

SERIES SYMBOLS OR IMAGES

`Line`, `Line Bar`, `X-Y`, `Time Plot`, `Pareto`, and `Radar` support custom symbols or images to mark data items in a series. By default, symbols are enabled for `Time Plot`, `X-Y`, and `Radar` graphs. They are disabled for `Line` and `Pareto` graphs.

Specify symbols and images on a series-by-series basis. The default symbol is the `Rectangle`, but there are several different symbols from which to choose.

Properties and attributes related to series symbols are available from the `Symbol Settings` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

Note: Symbols must be enabled for this property to be visible in `Object Properties`.

Warning: If you disable the symbol for a series and do not enable a line, area, or bubble, the series is rendered invisible.

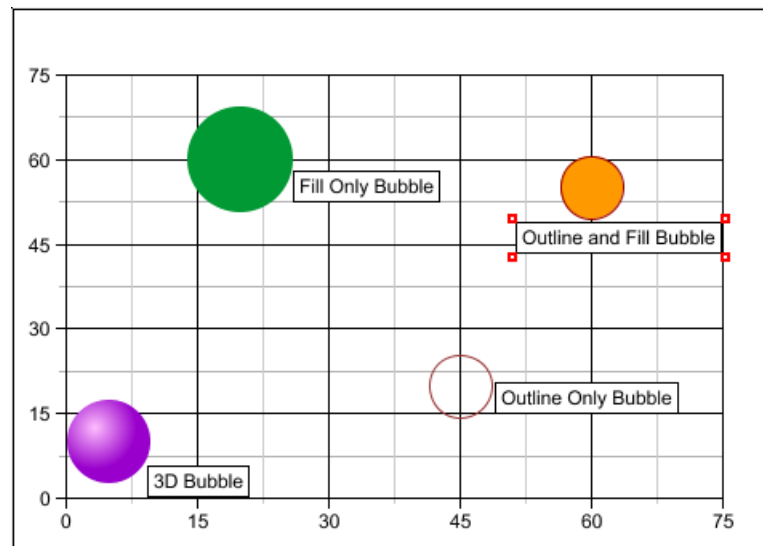
▪ GRAPHS

▪ *Line-Based and Plot Graphs*

BUBBLE COLOR AND SIZE

The **X-Y bubble**, **X-Y combo**, and **Time bubble** graph sub-types support bubbles to mark data points in the graph.

Bubbles reflect a third data value for a data item, but this value must be specified for the bubble to appear. For more information about bubble graphs, see Chapter 15, “[Bubble Graphs](#),” in the *Corda 7 Graph Guide*. By default, bubbles are enabled for bubble graphs.

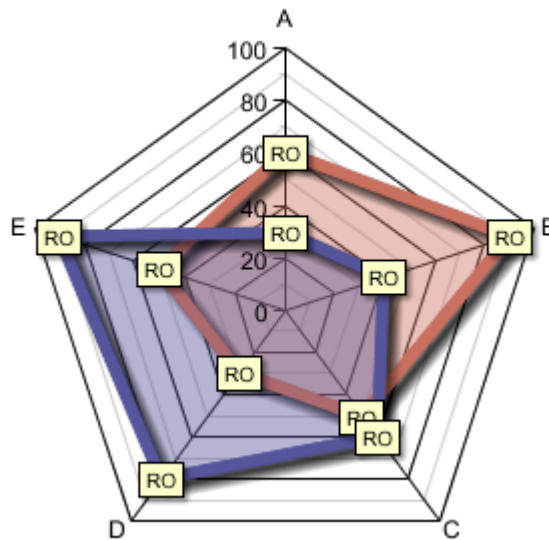


By default, the bubble data value sets the diameter of the bubble, but it can be configured to set the area of the bubble; or scale the bubble size to be larger or smaller than the bubble value (useful for very small or very large numbers). Sizing options are set for the entire graph.

Properties and attributes related to bubble color and size are available from the **Bubble Settings** property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

RADAR GRAPH TRANSPARENCY

To help make things more readable, the filled area for each data series in a radar graph is partially transparent. A transparency value lets you specify the degree of transparency: *100* makes the filled areas transparent; *0* makes the filled area solid. The default setting is *75*.



Properties and attributes related to radar graph transparency are available from the [Radar Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

▪ GRAPHS

▪ Heat Maps

HEAT MAPS

Heat maps use colored rectangles to represent data values. Rectangle colors vary depending on how a particular data point fits into the specified data ranges. This section discusses the following topics:

- [Data Ranges](#)
- [Fill](#)
- [Margins and Grid Lines](#)

DATA RANGES

Data ranges are a map concept used with heat maps. For more information on data range concepts, see Chapter 7, “[Map Ranges](#).”

Properties and attributes related to heat map data ranges are available from the `Ranges` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

FILL

The color of individual heat map components is determined by the data range into which the component’s value falls.

Properties and attributes related to heat map fill are available from the `Bar Settings` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

MARGINS AND GRID LINES

There are two ways to separate the various fields that make up a heat map: margins and grid lines. *Margins* define the spacing between individual heat map fields. A larger margin necessarily causes each field in the heat map to shrink proportionately. *Grid lines* are separators between columns and rows of heat map fields.

Properties and attributes related to heat map margins and grid lines are available from the `Grid` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

GAUGES

Gauges are a special type of graph that depict a single data value, and classify that value as belonging to a predetermined range. Typically, a gauge is used to indicate the status of a data value and if that data warrants some kind of action. For example, a gauge can be used to indicate whether inventory is normal or too low or if the Web server load is extraordinarily high.

Although different gauges represent the gauge's value differently, all gauges require certain information, including a gauge name, one or more data ranges, and a scope for each data range (the minimum and maximum values for data items in a range). Some gauge types can also require color, shape, or image definitions. Once the gauge is created, assign a data value to the gauge, and the gauge determines to which range the value belongs and then uses the characteristics of that range to represent the data value.

In addition to representing the range into which a data value belongs, many gauges also display the other ranges defined for the gauge.

For Corda Builder, gauges are essentially groups of shapes to which specific gauge functionality is applied. As such, creating gauges relies heavily on Corda Builder's [Shape Tools](#). For more information about creating shapes, see ["Shapes" on page 8-6](#).

Corda Builder provides the following gauge categories: [Basic](#), [Linear](#), [Repeating](#), and [Radial](#). Use these gauge categories to create any type of gauge you might need. The [Corda 7 Gallery](#) includes gauges from each category. For more information on the Gallery, see ["The Corda 7 Gallery" on page 3-8](#).

This topic includes the following sections:

- [Gauge Labels](#)
- [Gauge Ranges](#)
- [Basic Gauges](#)
- [Linear Gauges](#)
- [Repeating Gauges](#)
- [Radial Gauges](#)
- [Disassembling Gauges](#)

- GRAPHS
- Gauges
-
-

GAUGE LABELS

All gauges, regardless of type, include a gauge label as part of the gauge object. Configure gauge labels as you would for any other object label; using [Object Properties](#) to configure the [Label](#) property.

Position the gauge label using the [Select a Point](#) tool (in the Tools view.) by clicking-and-dragging the label cell to the desired position.

Note: *You can place the gauge label anywhere on the **Project Canvas**. However, since the gauge label is part of the gauge object to which it is associated, positioning the gauge label outside the gauge shape automatically resizes the gauge object's boundaries to enclose the gauge label's new position.*

GAUGE RANGES


Gauge ranges are similar to map ranges in that they override the default gauge shape properties as needed to properly represent the data. Gauge ranges provide different types of overrides depending on the gauge type.

By default, any gauge has the following five ranges when it is created:

- Out of Range Low (<0.0)
- Green (0.0 - 50.0)
- Yellow (50.0 - 80.0)
- Red (80.0 - 100.0)
- Out of Range High (>100.0)

Gauges let you do the following range-related activities:

ADD A RANGE: Click the [Add](#) button  in the [Ranges](#) property.

DELETE A RANGE: Click the [Remove](#) button  in the appropriate [Range](#) property.

MODIFY A RANGE: Modify the characteristics of a range, including range minimum and maximum values, in the appropriate [Range](#) property.

Properties and attributes related to gauge ranges are available from the [Ranges](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

BASIC GAUGES

Basic gauges let you create gauges out of groups of one or more shapes. Basic gauges can be just about anything you want. Use combinations of shapes, and apply Ranges to those shapes in any way you want to achieve the desired gauge effect.



To create a Basic Gauge

Note: *The following steps are the minimum needed to create a basic gauge. More complex gauges may involve additional steps to create the desired effects.*

- 1 **Using Corda Builder's Shape Tools, create the desired gauge shapes.**

For more information about creating shapes, see "Shapes" on page 8-6.

- 2 **Position the gauge component shapes as needed to create the desired gauge look.**
- 3 **Select all the gauge's component shapes on the Project Canvas or in the Object List.**
- 4 **Right-click the component shapes and select Convert to Basic Gauge.**

Notice that the selected Shapes in the Object List are grouped into a single Gauge object.

- 5 **Configure the new freeform gauge as needed.**

Select the gauge and use Object Properties to configure the gauge.

BASIC GAUGE RANGES

Basic gauge ranges can override the following shape characteristics to represent gauge data:

Shape Visibility: Assign shapes to ranges, and display shapes based on the range into which the gauge value falls. This lets you vary the look of the entire gauge as the gauge value changes.

Shape Fill: Vary gauge fill properties based on the range into which the gauge value falls. Multi-shape gauges let you specify which shape(s) get range-based fill.

Shape Border: Vary shape border properties on the range into which the gauge value falls.

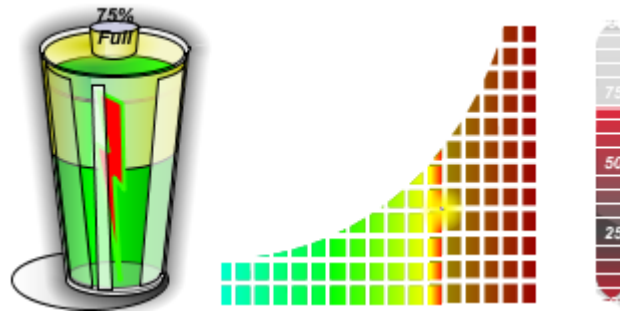
- GRAPHS
- Gauges
-
-

Shape Shadow: Vary shape shadow properties based on the range into which the gauge value falls.

Properties and attributes related to basic gauge ranges are available from the [Gauge Settings](#), and [Ranges](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

LINEAR GAUGES

Linear gauges are single-shape gauges that “fill up” to indicate the current gauge value. Common filled gauges include bar gauges and thermometers.



To create a Linear Gauge

Note: *The following steps are the minimum needed to create a linear gauge. More complex gauge shapes may involve additional steps to create the desired effects.*

- 1 **Using Corda Builder’s [Shape Tools](#), create the desired gauge shape.**

For more information about creating shapes, see [“Shapes” on page 8-6](#).

- 2 **Right-click the shape on the Project Canvas or in the Object List, and select [Convert to Linear Gauge](#).**

Notice that the object in the Object List changes from a Shape to a Gauge.

- 3 **Configure the new linear gauge as needed.**

Select the gauge and use [Object Properties](#) to configure the gauge.

LINEAR GAUGE RANGES

Linear gauge ranges can override the following shape characteristics to represent gauge data:

Shape Fill: Vary gauge fill properties based on the range into which the gauge value falls. Multi-shape gauges let you specify which shape(s) get range-based fill.

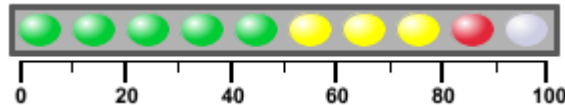
Shape Border: Vary shape border properties on the range into which the gauge value falls.

Shape Shadow: Vary shape shadow properties based on the range into which the gauge value falls.

Properties and attributes related to linear gauge ranges are available from the [Ranges](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

REPEATING GAUGES

Repeating gauges are two-shape gauges. One shape is the gauge frame and the other is the repeating shape used to indicate the current gauge value. The most common repeating gauge is an LED.



To create a Repeating Gauge

Note: *The following steps are the minimum needed to create a repeating gauge. More complex gauges may involve additional steps to create the desired effects.*

- 1 **Using Corda Builder's Shape Tools, create the desired gauge shapes.**

Create one shape for the gauge frame and one for the repeating shape. For more information about creating shapes, see ["Shapes" on page 8-6](#).

Note: *By default, Corda 7 automatically selects the larger of the two shapes as the gauge frame. However, after creating the gauge, you can change the shape designation, if desired.*

- 2 **From the Project Canvas or the Object List, select both component shapes; then right-click the shapes and select Convert to Repeating Gauge.**

Notice that the two Shapes in the Object List change to a single Gauge object.

- 3 **Configure the new repeating gauge as needed.**

Select the gauge and use Object Properties to configure the gauge.

- GRAPHS
- Gauges
-
-

REPEATING GAUGE RANGES

Repeating gauge ranges can override the following shape characteristics to represent gauge data:

Shape Fill: Vary gauge fill properties based on the range into which the gauge value falls. Multi-shape gauges let you specify which shape(s) get range-based fill.

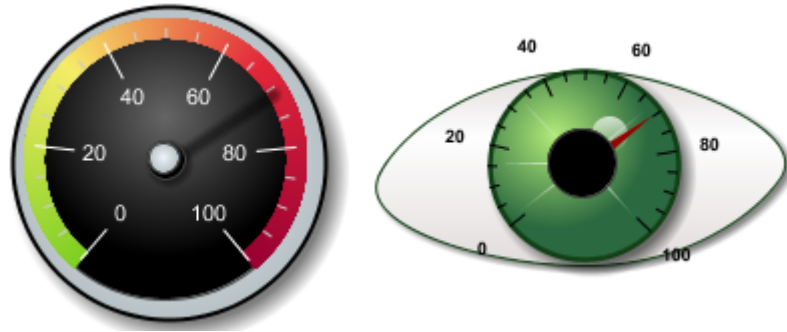
Shape Border: Vary shape border properties on the range into which the gauge value falls.

Shape Shadow: Vary shape shadow properties based on the range into which the gauge value falls.

Properties and attributes related to repeating gauge ranges are available from the [Ranges](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

RADIAL GAUGES

Radial gauges let you create dial gauges with pointers that indicate the current gauge value. Common repeating gauges include dials and speedometers.



To create a Radial Gauge

Note: *The following steps are the minimum needed to create a simple radial gauge. More complex gauge shapes may involve additional steps to create the desired effects.*

- 1 **From Corda Builder's Shape Tools, select Create a dial gauge shape.** 

For detailed information about creating a dial shape, see [“dial Gauge Shape” on page 8-12](#).

The dial shape is a framework for the gauge action and does not accept fill or other dial face properties.

- 2 **(Optional) To customize the look of the dial shape, create a second shape of identical size and overlay it on the dial shape. Modify the look and feel of the overlay shape to achieve the desired dial effect.**

Remember that shapes adhere to Corda 7 layer rules (the first shape is the backmost layer, with subsequent shapes layered on top of the first layer in order of creation).

- 3 **Create the desired gauge indicator shape(s).**

The gauge indicator is the spindle of a radial gauge. It can be any combination of shapes. Configure (fill, border, shadow, etc.) and align all indicator shapes to create the desired spindle look and feel.

Important: *Lay out the indicator shapes vertically, with the pivot end of the indicator at the bottom and the pointer end of the indicator at the top. Corda Builder assumes this orientation when creating the indicator object.*

- 4 **From the Project Canvas or the Object List, select all the indicator shapes. Then right-click and select Create gauge indicator.**

Corda Builder creates two handles on the spindle. These handles can be moved vertically along the spindle's center axis and do not have to be placed within the spindle shape.

- *Pointer handle (yellow):* Defines the pointer end of the spindle that appears closest to the gauge scale. With the [Select a Point](#) cursor from [Shape Tools](#), drag the Pointer handle to any desired location above the spindle's Pivot handle.
- *Pivot handle (blue):* Defines the pivot point on the spindle. Corda Builder aligns this handle with the pivot point in the dial shape when creating the radial gauge. With the [Select a Point](#) cursor from [Shape Tools](#), drag the Pivot handle to any desired location at or below the spindle's Pointer handle.

- 5 **From the Project Canvas or the Object List, select the dial shape and the indicator shape, and then right-click and select Convert to Radial gauge.**

Corda Builder assembles the various shapes into a single Gauge shape and automatically creates a gauge scale.

- 6 **Configure the new radial gauge as needed.**

Select the gauge and use [Object Properties](#) to configure gauge properties.

- **GRAPHS**
- *Gauges*
-
-

RADIAL GAUGE RANGES

Radial gauge ranges can override the following shape characteristics to represent gauge data:

Shape Fill: Vary gauge fill properties based on the range into which the gauge value falls. Multi-shape gauges let you specify which shape(s) get range-based fill.

Shape Border: Vary shape border properties on the range into which the gauge value falls.

Shape Shadow: Vary shape shadow properties based on the range into which the gauge value falls.

Properties and attributes related to radial gauge ranges are available from the [Ranges](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

DISASSEMBLING GAUGES

Since gauges are just collections of shapes, they can be disassembled into their component shapes if desired. This makes it possible to re-use gauge components.

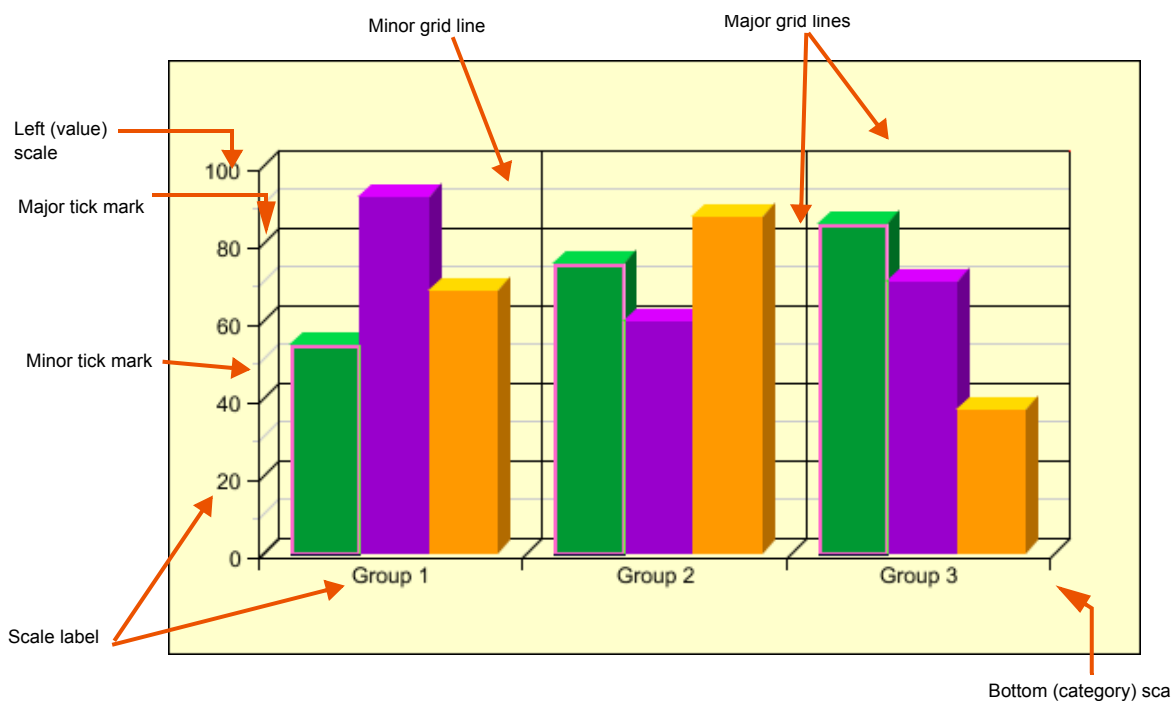
To disassemble a gauge, right-click the appropriate gauge object on the [Project Canvas](#) or in the [Object List](#), and select [Ungauge](#).

SCALES AND GRIDS

Graphs essentially create a way of visualizing two or more characteristics of the associated data set. For example, a graph might illustrate the date and numeric value of a data item, or depict quantities of data items that fall into a specified category, such as *Expenditures* or *Revenues*.

Scales and grids help visually clarify a graph's data. *Scales* are located along the sides and bottom of a graph, and use a combination of tick marks and labels to indicate the category, time period, or numeric value of each data item in the graph.

Grids are extensions of scale tick marks and labels, and consist of a number of evenly spaced vertical or horizontal lines in a graph's background. Use them like a ruler to judge the value of data items in the graph.



Note: Scales and grids are not used in pie graphs.

· SCALES AND GRIDS
·
·
·
·

The first few sections of this chapter help you conceptually understand how scales and grids work:

- [Terminology](#)
- [Scale Types](#)

The next sections of this chapter deal with each type of scale individually:

- [Value Scales](#)
- [Category Scales](#)

The last sections discuss scale and grid features common to all graphs, regardless of the scale types they use:

- [Synchronizing Scales](#)
- [Grids](#)
- [Tick Marks](#)
- [Scale Markers](#)
- [Scale Titles](#)



TERMINOLOGY

The following definitions are helpful when you read this chapter:

AXIS Each graph (except pie and radar) has two *axes*—a horizontal axis (x-axis) and a vertical axis (y-axis).

The horizontal axis represents the baseline, or 0 value, for the side scale. It is typically the bottom-most horizontal line, although in graphs with negative values, it might appear somewhere in the middle of the graph.

The vertical axis has different meanings depending on the bottom scale type. For time scales, it represents the start date of the scale. For value scales, it represents the baseline or 0 value. For category scales, it has no special meaning. It is typically the left-most vertical line, although in graphs with negative values, it might appear somewhere in the middle of the graph.

The term *axis* is also used generically to refer to the left-most vertical line or the bottom-most horizontal line. Thus, the side scale is said to appear along the vertical axis, and the bottom scale is said to appear along the horizontal axis, even if the axis, technically speaking, is in the middle of the graph.

GRID LINES Lines that extend from the tick marks along an axis across the entire background of the graph. They are parallel to the opposite axis of the graph. Grid lines help make visual comparisons of data easier, serving as a point of reference for the scale they extend from.

There are both *major* and *minor* grid lines, corresponding to major and minor *tick marks*. Typically, major grid lines are heavier and darker than minor grid lines.

RANGE The numeric extent of the values covered by a value scale. A range spans from a minimum value (often 0) to a maximum value. If a data item is outside the range of a value scale, it is not graphed.

SCALE A system of ordered marks at fixed intervals that are used to categorize or classify the data in the graph. Each scale classifies one characteristic of the data, such as the date or numeric value. In most scales, the classification is progressive, each interval representing a growth of size, value, or time. Most graphs have two scales, one along the vertical axis and one along the horizontal axis. Scales are indicated by labels and tick marks along these axes.

There are several different types of scales, each suitable for classifying different data characteristics. These include *value scales*, *category scales*, *time scales*, and *time category scales*. These are defined in the next section.

SCALE LABELS Text appearing beside or below each tick mark and grid line that describes the meaning of the scale at that point (e.g., *50* or *11/25/02* or *Marketing*). It essentially classifies data items that appear along the indicated grid

- SCALES AND GRIDS
- Terminology
-
-

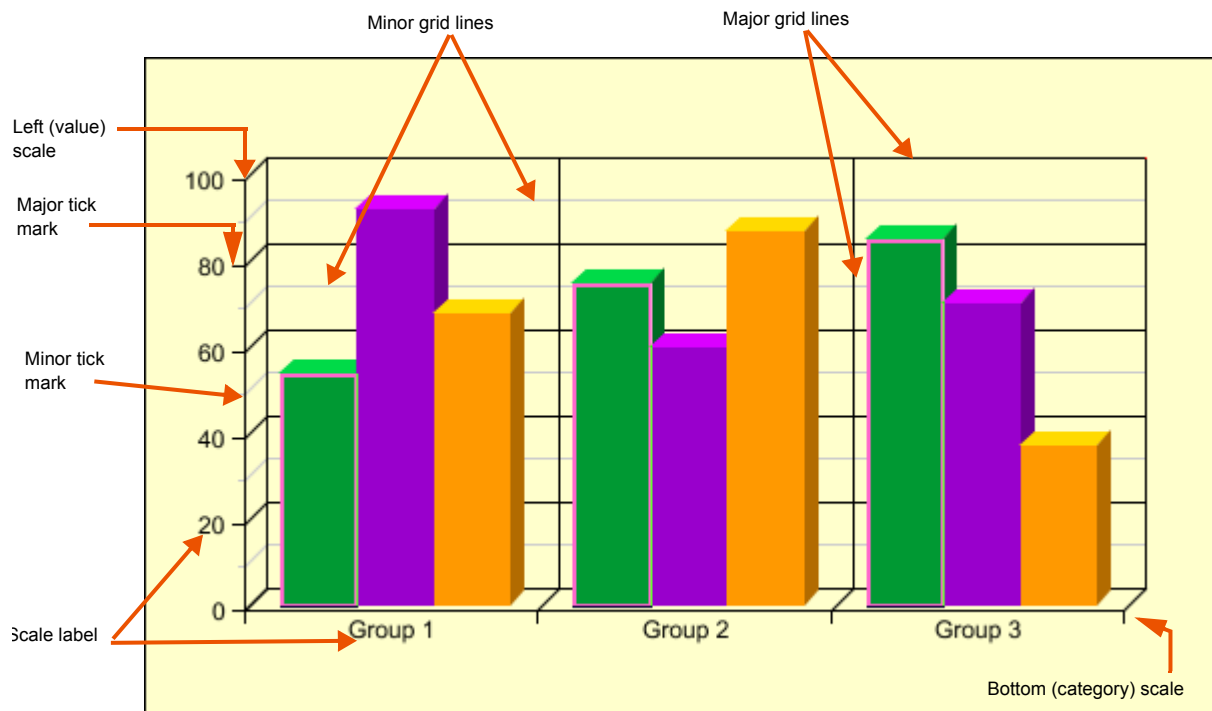
line. Scale labels always appear next to *major tick marks*. In time scales, smaller labels might accompany *minor tick marks*.

SCALE MARKER A line or colored region in front of or behind a graph that extends from an axis to the other end of the graph. Scale markers are very similar to grid lines or grid areas, both in looks and purpose. However, they are typically a different color than the grid lines or grid areas. They can be placed anywhere you want, whereas grid lines and grid areas only occur in automatically determined areas. Also, the user must create scale markers manually.

Scale markers should be used when you want to highlight or call attention to data items that fall within a certain range or above or below a certain value. They can be used in value scales and times scales but not in category or time category scales.

TICK MARKS Small line segments of measurement that intersect an axis, indicating specific values along the scale.

There are two types of tick marks: *major* and *minor*, corresponding to major and minor *grid lines*. Major tick marks are larger and more pronounced than minor ones and are labelled with a *scale label*. Minor tick marks occur between each major tick mark.



SCALE TYPES

Four types of scales are used in Corda Builder graphs. Each scale has unique formatting options. Select the header of each scale description to navigate to a more detailed explanation of each scale type.

CATEGORY SCALES Used to indicate the category to which a data item belongs. It is usually placed along the horizontal axis (and thus often referred to as the top or bottom scale). Each tick mark and label along the scale represents a unique category. This is a discrete scale, meaning that there are no in-between values—a value must belong to one category or another.

TIME CATEGORY SCALES Used to indicate the date and time of the data items. This scale can be used in place of a *category scale* in any graph that supports *category scales*. In fact, the only real difference between the two scale types is that *time category scales* treats category names as dates.

A *time category scale* differs from a *time scale* in that it is discrete, meaning there are no in-between values—a data item must belong to one of the dates shown below in the scale. It cannot occur between any of these dates (if it did, the graph would create a new scale label to accommodate it).

TIME SCALES Used to indicate the date and time of the data items. This scale, used only for time plot graphs, is placed along the horizontal axis. It is based on date or time, as opposed to numbers.

A *time scale* differs from a *time category scale* in that it is continuous, and not confined to discrete categories. For example, the time scale might only label *May 1st*, *May 16th*, and *June 1st*, but it can still plot a date like *May 10th* by putting the data item somewhere in between *May 1st* and *May 16th*.

VALUE SCALES Used as a number-based scale with all graphs except pie charts and heat maps. Typically, this scale is along the vertical axis of a graph, although it can occur along the horizontal axis.

In X-Y graphs, there are two value scales. The *X scale*, which depicts *x* values, occurs along the horizontal axis. The *Y scale*, which depicts *y* values, occurs along the vertical axis.

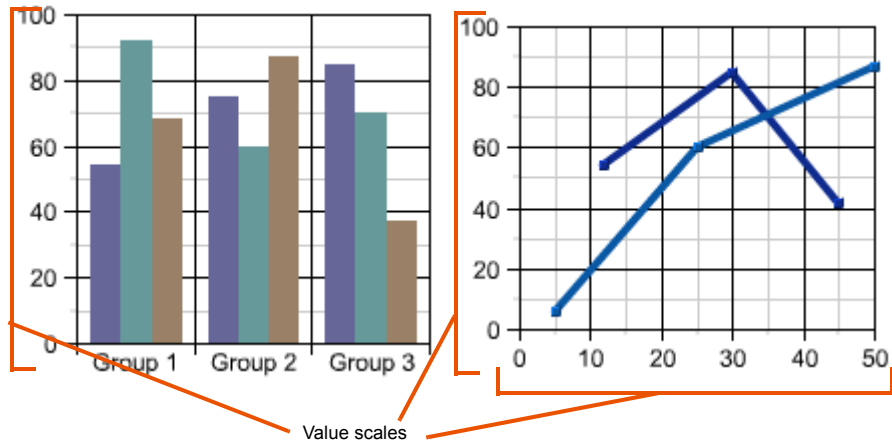
A *Value scale* is continuous and not confined to discrete categories. For example, the value scale might only label *100*, *150*, *200* and so on, but it can still plot a value of *131* by putting the data item somewhere in between *100* and *150*.

SCALES AND GRIDS

Value Scales

VALUE SCALES

Value scales are used to indicate the numeric values of the graph's data items. All graphs, except for pie graphs and heat maps, have at least one value scale. X-Y graphs have two value scales—one for both the *x* and *y* axes. Likewise, line bar combo graphs and graphs with dual *y* scales have two value scales.

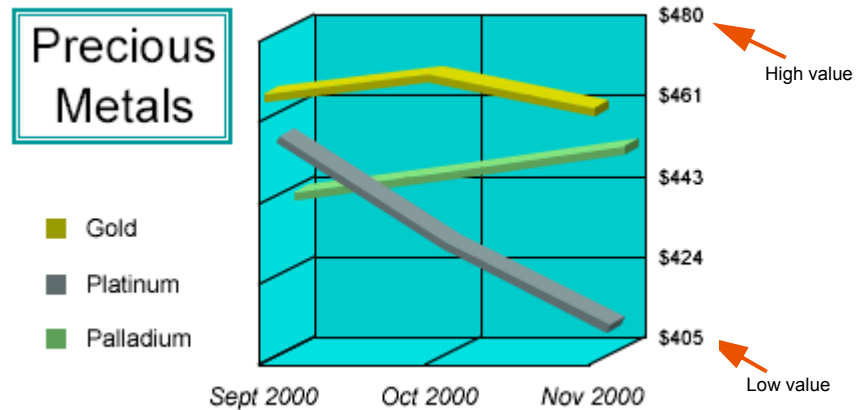


There are a number of customizing features for value scales, and each is described in the following sections:

- [Value Scale Range](#)
- [Value Scale Labels](#)
- [Logarithmic Scales](#)
- [Value Scale Position](#)
- [Dual Value Scales](#)

VALUE SCALE RANGE

The range of the scale is, by definition, the set of values that the scale encompasses. This is defined by a minimum (low) value, and a maximum (high) value. In the graph below, the range is from 405 to 480.



By default, Corda Builder automatically computes the value scale range to include all data items, so defining the value scale manually is typically unnecessary.

For example, if the data items are 2, 6, 9, and 3, the graph automatically determines the range of its scale to be 0 to 10. On the other hand, if the values are 289, 390, and 457, it automatically determines the range to be 0 to 500.

Note: *Corda Builder attempts to use round numbers when creating ranges. Thus, the latter range is 0 to 500 instead of 0 to 457.*

A graph's value scale can be configured in four ways:

AUTOMATICALLY Leaves configuration of the graph's value scale to Corda Builder, which configures the value scale as described above.

CUSTOM Lets you specify the rules that define how a graph creates its value scale.

MULTIPLES-OF Lets you configure major and minor scale values as multiples of a specified starting number.

MANUALLY Lets you configure a graph's value scale manually. The disadvantage of manually scaling is that it makes the graph less flexible to data changes.

▪ SCALES AND GRIDS

▪ Value Scales

Some of the formatting options you should consider for graph value scales include the following:

- [Value Scale Base](#)
- [Value Scale White Space](#)
- [Value Scale Divisions](#)

VALUE SCALE BASE

By default, Corda Builder includes zero in the value scale as the base (bottom line) of the scale, but this is not required. Having a value scale base other than zero might be useful when data items are grouped, but not very close to zero. Corda Builder lets you choose to include zero in the value scale with [automatic](#), [custom](#), and [multiples-of](#) value scale settings. [Manual](#) graph scale settings explicitly define a value scale base.

Warning: *Setting the value scale base to other than zero can deceive a viewer of the graph because most viewers assume that the base of a graph is inherently zero.*

Properties and attributes related to the value scale base are available from the [Divisions](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

VALUE SCALE WHITE SPACE

By default, Corda Builder inserts white space between the highest data value and the top of the scale; both to improve graph appearance, and because Corda Builder attempts to use round numbers (50 as opposed to 47) for the scale limits.

Note: *When setting the value scale base to something other than zero, the amount of white space below the lowest data item is configurable.*

Corda Builder lets you configure graph white space when using [custom](#) value scale settings. White space is expressed in terms of the percentage of space between the base of the graph and the highest data item. So, for example, in a graph where the highest data item is 50 and the base is zero, setting this value to 10 percent means that the white space adds 5 to the scale. Changing the value scale size applies only to automatically scaled graphs. Manually scaled graphs explicitly define the value scale size.

Note: *Even after adding white space, Corda Builder still rounds the graph's value scale to an even number, so this number may not exactly reflect the resulting graph white space.*

Properties and attributes related to the value scale white space are available from the [Divisions](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

VALUE SCALE DIVISIONS

Corda Builder divides graph scales into a certain number of divisions, each of equal value (except in logarithmic scales). Each major division has a scale label, a tick mark, and a grid line. Minor divisions divide the gap between each major division.

For example, a scale with a range of 0 to 500 is divided into five major divisions, meaning there are six labels: 0, 100, 200, 300, 400, and 500. Between each label is a minor tick mark, indicating 50, 150, 250, 350, and 450.

Control the number of labels, grid lines, and tick marks in a value scale by adjusting the number of divisions in the scale.

While the graph scale won't have more divisions than specified here, it can have fewer since Corda Builder attempts to use round numbers in its scale labels. For example, if using seven divisions creates labels like 16.67, 33.33, etc., Corda Builder might use six divisions with round labels like 20, 40, etc.

Corda Builder lets you control the major and minor divisions in any of the value scale settings ([automatic](#), [custom](#), [multiples-of](#), and [manual](#)).

Properties and attributes related to the value scale divisions are available from the [Divisions](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

VALUE SCALE LABELS

Next to each major tick mark or grid line in a value scale is a scale label. This label is simply a number that indicates the value of the scale at the indicated tick mark or grid line.

The number of labels in a value scale is dictated by the number of major divisions in that scale. (See [“Value Scale Divisions” on page 5-8](#) for information about adjusting the number of divisions in the scale.)

Corda Builder lets you customize the following aspects of the scale labels:

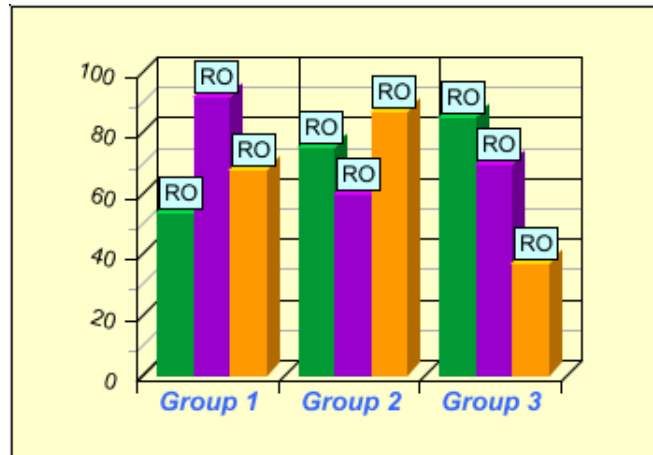
LABEL VISIBILITY By default, scale labels are visible, but in some situations it makes sense hide those labels. Reselect it to make scale labels visible again.

LABEL FONT Change the font style and color for the scale labels to fit the desired style of the graph. This dialog lets you configure the font, font style, font size, and font color.

- SCALES AND GRIDS

- Value Scales

LABEL ROTATION Rotate scale labels to make them more readable, or to decrease the space required to display the scale labels. Choose a rotation of 15, 30, 45 degrees, either clockwise (negative value) or counter-clockwise (positive value).



LABEL NUMERIC FORMATS AND ABBREVIATIONS There are times when scale labels need to be manipulated to make them more readable, or to convey additional information beyond the simple number. For example:

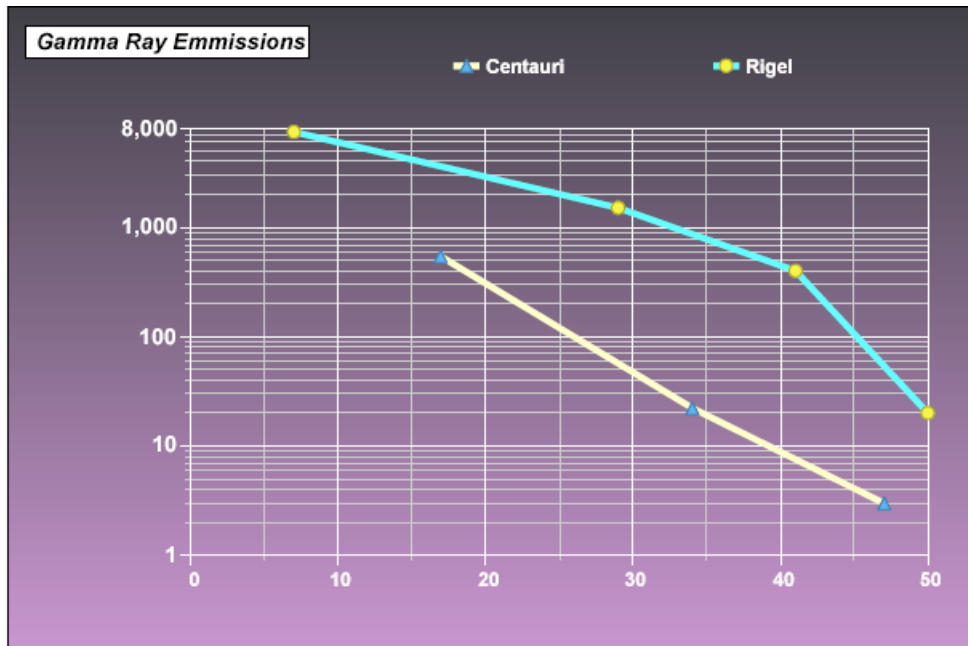
- A scale label that displays large numbers might need to be modified to display numbers using an abbreviation. For example, *1,250,000*, might be abbreviated as *1.25m*.
- If a scale label's values are percentages, meaning a value of 40 represents 40%, the scale labels should be configured to display appropriately.

Properties and attributes related to the value scale labels are available from the [Labels](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

LOGARITHMIC SCALES

In most scales, the increment between each tick mark is of equal value (e.g., 10, 20, 30, 40, and so on). In logarithmic scales, the increments between tick marks increase in a logarithmic fashion (e.g., 10, 100, 1000, 10000, and so on).

Logarithmic scales make it possible to better graph widely spaced data sets, or data sets with with extreme outliers. Any value scale can be a logarithmic scale.



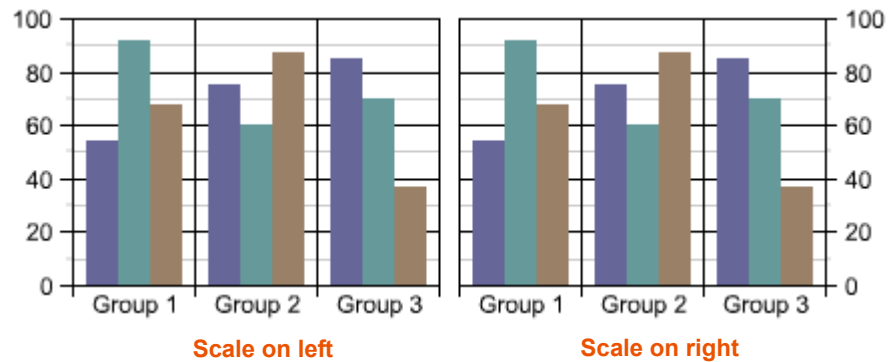
Properties and attributes related to the value scale divisions are available from the [Divisions](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- SCALES AND GRIDS
- Value Scales
-
-

VALUE SCALE POSITION

The value scale on the graph's vertical axis can be on either the left side or right side of the graph (left is default).

Note: *This does not apply to dual-y scale graphs.*

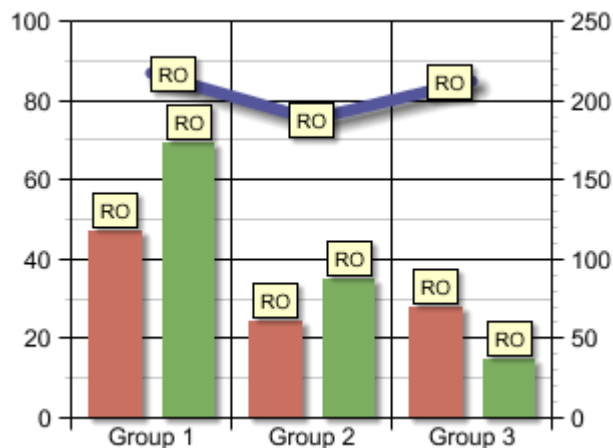


Properties and attributes related to the value scale position are available from the `Value Scale` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

DUAL VALUE SCALES

Line-bar combo graphs have separate value scales on each side of the graph. By default, the left-side scale is for the line data series, and the right-side scale is for the bar data series.

Each scale operates independently of the other, but the formatting options are the same. This allows you to graph two disparate data sets on the same graph.



However, it is possible to synchronize the scales, if desired. This creates a new scale that encompasses the range of all data values in the graph.

For example, consider a line-bar graph with line data series values that range from 0 to 100, and bar data series values that range from -20 to 90. With a maximum of 5 major increments, the resulting line scale labels display 0, 25, 50, 75, and 100, while the bar scale labels display -20, 7.5, 35, 62.5, and 90.

When synchronized, the resulting scale labels are -50, 0, 50, and 100, and they can be used on both sides of the graph. Hide the right-side scale when scales are synchronized, if desired. For additional information on setting scale values, see [Value Scale Range](#).

Corda Builder lets you swap the line scale with the bar scale so that the line scale is on the right and the bar scale is on the left.

Properties and attributes related to dual value scales are available from the [Value Scales](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- SCALES AND GRIDS

- Value Scales

TIME SCALES

Time scales are [Value Scales](#) that present the associated scale values as dates and/or times. Time scales are used in the time plot graphs. There are several ways to customize a time scale and the way it displays times and dates.

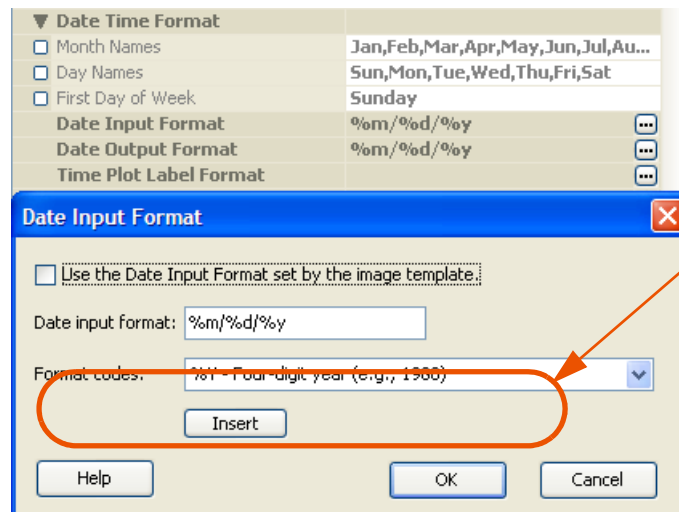
Time scales support the same configuration properties available to other types of value scales, in addition to some properties specific to time scales.

DATE FORMAT

Each time plot graph has a *date input format* string that defines how the graph interprets date/time input. If you encounter a problem with times and dates at the bottom of the graph not displaying correctly, it is typically because the date input format doesn't match the format of the date/time data passed into the graph.

Modify the date input format string so that the graph can accept date/time data in almost any desired format. The date input format string uses meta tags to indicate where in the time values a graph should expect to find the different date/time components, such as the month, day, or minute. Include standard text separators (such as slashes or colons) to organize the display of date/time data, if desired.

For example, the default date input format string is `%m/%d/%y`. This means that when you enter data into the [Data Editor](#), or when you send data dynamically to Corda Server, enter the month, followed by a slash, followed by the day, followed by a slash, followed by the two-digit year (e.g., 12/31/01).



Similarly, each time plot graph has a *date output format* string that defines how the graph displays date/time values in data labels when using the `%_TIME_VALUE` meta tag in the [Data Labels](#) property. Modify the date output format as needed to display date/time values in the correct format. The default setting is `%m/%d/%y`.

A complete list of date/time meta tags is available in [Table 5.2](#).

TABLE 5.2 Date-Formatting Meta Tags

%Y	Four-digit year (e.g., 2001)
%y	Two-digit year (e.g., 01)
%m	Month of year (1-12)
%b	Name of month
%d	Day of month (1-31)
%H	Hour (0-23)
%M	Minute (0-59)

Note: *The `%b` meta tag displays month data as the month name (e.g., January, February, etc.). In some cases, you might want to define what month names a graph should expect to find in the dates. For example, if all of the month names are abbreviated, tell the graph to look for these abbreviations instead of the full month names.*

Properties and attributes related to the date format are available from the [Date Time Format](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

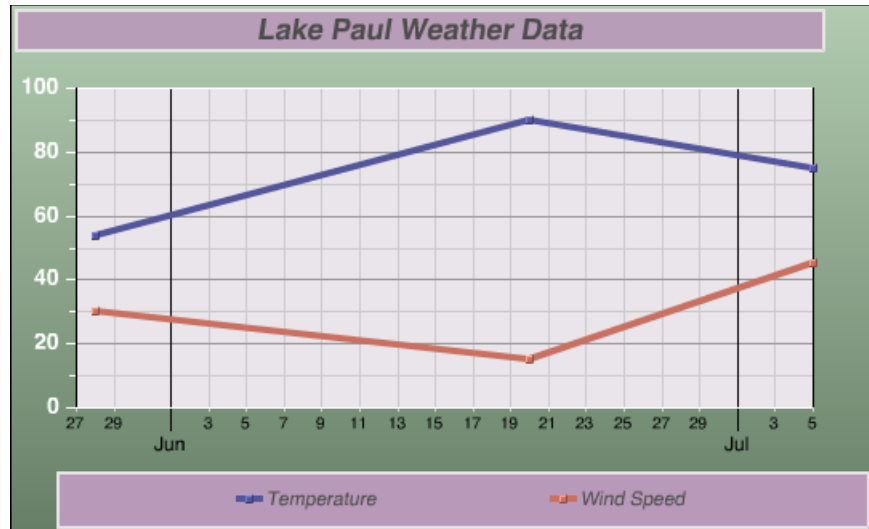
TIME SCALE BEHAVIOR

By default, the time scale automatically sizes itself to accommodate the range of data items it needs to plot. As part of this sizing algorithm, the time scale determines a scale increment for the scale (see [“Value Scale Divisions” on page 5-8](#)). The resulting scale increment defines the distance between each major tick mark and scale label.

Time scales are unique in that they also can have minor scale labels. Minor scale labels are automatically applied when there are only one or two major scale labels. For example, if the time plot graph’s data items span a single month, the month is still the major scale label.

- SCALES AND GRIDS
- Value Scales
-
-

However, since that label doesn't provide sufficient information, the graph displays minor scale labels indicating days of the month.



While the algorithm that graphs employ to determine the increments between major and minor tick marks and labels is complicated, similar controls are available for time scales as for other value scales (see ["Value Scales"](#) on page 5-6).

Properties and attributes related to time scale behavior are available from the `Time Value Scale` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

CATEGORY SCALES

The labels that appear along the category scale (usually along the x-axis) of most graphs represent category names. Each category of data in the graph has a label. Change these labels by editing the category names in the [Data Editor](#) (refer to “[Column and Row Names](#)” on page 4-11).

Note: *Pie, X-Y, and time plot graphs do not have category scales. Heat maps, on the other hand, have category scales on both the horizontal and vertical axes.*

Corda Builder provides the following category scale actions:

- [Category Scale Position](#)
- [Category Scale Labels](#)

CATEGORY SCALE POSITION

The default location of a category scale on the graph’s horizontal axis is below the graph, but Corda Builder lets you move it above the graph, if desired.

Properties and attributes related to the category scale position are available from the [Category Scale](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

CATEGORY SCALE LABELS

The labels that appear along the category scale (usually along the x-axis) of most graphs represent category names. Each category of data in the graph has a label.

CATEGORY SCALE LABEL BORDERS

Configure a border around each category label to help distinguish between them in the graph. There are two options for doing so: use full borders around each category scale label, or use dividers between category scale labels. In either case, Corda Builder uses the defined label borders until label adjustments make it impractical to do so.

Properties and attributes related to the category scale labels are available from the [Category Scale](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- SCALES AND GRIDS

- *Category Scales*

CATEGORY SCALE LABEL FONT

Configure the look of the font used with category scale labels, including font, font style, font size, and font color.

Properties and attributes related to the category scale font are available from the [Category Scale](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

AUTOMATIC LABEL ADJUSTMENTS

A graph can have a virtually infinite number of categories, so there might not be enough room to properly display labels for all of the data categories. When this happens, the graph uses an algorithm for finding ways to display as many labels as possible while still maintaining readability. This algorithm is known as *automatic label adjustment*.

Control the automatic label adjustment by specifying rules and actions the graph follows when category scale labels overlap.

When overlap occurs, the graph attempts to resolve the overlap by applying each selected rule and action, in order from top to bottom, until a rule or action is found that eliminates the overlap or it has exhausted its options.

After the graph has exhausted all individual options, it then tries combinations of rules, beginning with the first and second rule, then the first and third rule, and so on, until it finds a combination that works. The supported list of rules and actions includes the following:

Wrap Text To Max Number of Lines When selected, the text in the scale label is wrapped to the specified number of lines (if needed).

Shrink Font To If this rule is applied, the graph attempts to shrink the font of the scale labels down to the font point size specified. This rule is only applied as part of an automatic label adjustment.

Stagger Labels If this rule is applied, the graph attempts to stagger the labels so that every other one appears on an alternate level. This rule is only applied as part of an automatic label adjustment.

Rotate Labels If this rule is applied, the graph rotates the scale labels by the specified number of degrees.

Rotate Wrap Width Specifies the width, in pixels, of a rotated label.

Skip x Labels At a Time This rule specifies that a certain number of labels are skipped altogether. The bar, line, area, or plot points are still shown for that category, but the scale label is not. For example, a value of *1* displays every other label. If the value is *2*, it displays every third label. A value of *0* displays every label.

Properties and attributes related to automatic label adjustment are available from the `Adjust Labels` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

MANUAL LABEL ADJUSTMENTS

Manually adjust the scale label layout if a graph's automatic label adjustments don't work as needed.

Note: *Some of the options discussed in this section can be used in addition to automatic label adjustments; other options require you to turn off automatic label adjustments.*

Warning: *Disabling automatic scale adjustments in a project that is used as an Image Template file with Corda Server™ might result in overlapped labels, illegible graphs, or unnecessary abbreviations since the manual scale label adjustments are not adjusted for changing category scale label values.*

Label Length One of the first manual adjustments you might want to make is limiting the length of the label. `Corda Builder` lets you set separate limits for horizontal and rotated labels.

Staggering Labels Staggering labels causes the labels to alternate between a higher and lower level. This creates a somewhat larger label region to compensate for two levels of labels.

Label staggering can be configured with either automatic or manual label adjustments.

Rotating Labels By default, graphs display category scale labels horizontally. If necessary, rotate the labels so that they are at an angle or even vertical. This creates a somewhat larger label region to compensate for the rotated text.

Configure label rotation with either automatic or manual label adjustments. Select the number of degrees to rotate the label.

Rotate Wrap Width Specifies the width, in pixels, of a rotated label.

Skipping Labels If label space is limited, have the graph skip a certain number of category scale labels. The bar, line, area, or plot point is still shown for every category, but the label is excluded.

Label skipping can be configured with either automatic or manual label adjustments. The graph uses the number in the box to determine the increment between each shown bottom label. For example, if the value is `1`, it displays every other label. If the value is `2`, it displays every third label. If the value is `0`, it displays every label.

Note: *You can also specify the first category label to display, and skip one or more of the first category scale labels.*

SCALES AND GRIDS

Category Scales

Fixed Label Areas Heat maps support fixed areas within which the category scale labels are placed on the y-axis. The labels are left-justified within the defined area and automatically wrap around if necessary.

Properties and attributes related to manual label adjustment are available from the [Adjust Labels](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

TIME CATEGORY SCALES

In categorical graphs (e.g., bar graphs and line graphs), the data categories often represent periods of time. When this is the case, take advantage of time category scales to enhance the way the data is displayed.

Time category scales make categorical graphs look more like time plot graphs. A major scale indicates major intervals of time (such as months or quarters), and a minor scale indicates smaller periods of time (such as days). The time category scale is managed automatically, using an algorithm similar to that used with time value scales (see [“Time Scale Behavior” on page 5-15](#)).

Note: *Although time category graphs and time plot graphs may look similar, they are very different in the data they display. Time plot graphs should be used to plot data that occurs at infrequent intervals; time category graphs should be used for data that occurs at frequent intervals (daily, weekly, each business day, and so on).*

When using time category scales, remember the following concepts:

- Any graph with a category scale can use a time category scale, but instruct **Corda Builder** to change the scale to a time category scale. Time category scales use the same date input format string as time value scales (see [“Date Format” on page 5-14](#)).
- Specify the Major label format according to the way you want the data displayed. For example, `%b` shows the month on the major tick mark label.
- Specify the frequency of minor labels (either **Day** or **Week**).
- Specify the minor label format according to the way you want the data displayed. For example, `%d` shows the numerical date on the minor tick mark label.

In [Object Properties](#), properties and attributes related to time category scales are available from the [Category Scale](#) property, but only when [Use Time Category Scale](#) is selected in [Graph Settings](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

SYNCHRONIZING SCALES

When there are multiple graphs in an Image Template file, it sometimes makes sense to synchronize the scales between the graphs to better emphasize the relationships between them. Corda Builder provides this capability if the following prerequisites are met:

- Graph scales must be of the same type. For example, Corda Builder cannot synchronize a standard value scale with a time scale.
- The graph scales must represent the data in the same way. For example, Corda Builder cannot synchronize scales between a regular bar graph and an area graph, since the bar graph scale displays individual values and the area graph displays cumulative values.

To synchronize scales of multiple graphs, select the graph for which you want to synchronize scales; then <Shift>-right-click one of the other graphs and select **Synchronize graph scales**.

Note: *Synchronizing graph scales does not affect the look or format of the selected graphs in any way other than synchronizing the scales. Grid lines, grid stripes, colors, etc. are not affected by the scale synchronization.*

GRIDS

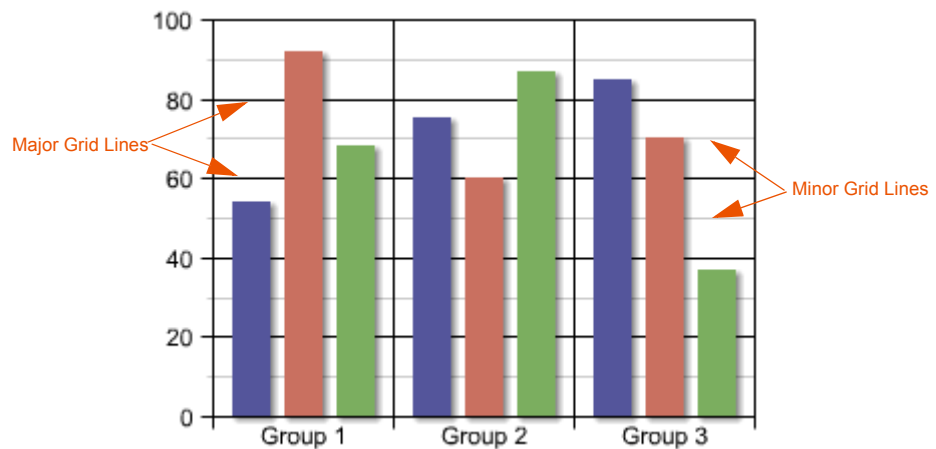
Many graph types include a *grid* behind their data items to help viewers understand the significance of the data. By functioning as an extension of the graph's scale labels and tick marks, grids help make the data easier to read.

Customize grids according to the graph's presentation needs. Corda Builder provides the following grid-related actions:

- [Grid Lines](#)
- [Background Colors](#)

GRID LINES

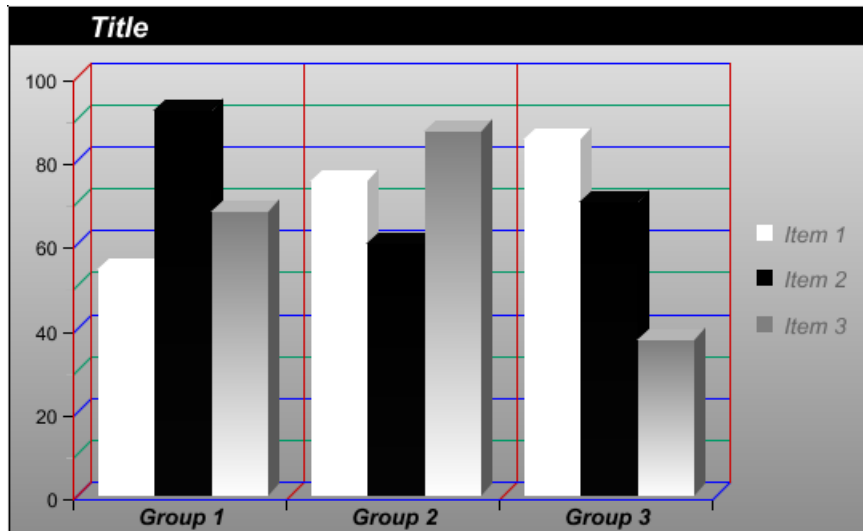
Grid lines are horizontal and vertical lines that extend across the background of the graph. The number of grid lines in the graph is determined by the number of increments in the scale (or the number of categories in the case of category scales). See "[Value Scale Divisions](#)" on page 5-8 for details. Corda Builder places grid lines at the outer extremes of the graph scale and between each major increment.



While you cannot manually add or remove a single grid line, add or remove grid line groups (such as minor or major groups) to make the graph more readable. Edit major and minor grid line settings in the [Grid](#) property in [Object Properties](#).

GRID LINE COLOR

By default, Corda Builder displays major grid lines in black and minor grid lines in light gray. However, changing the grid line color can help enhance graph readability.

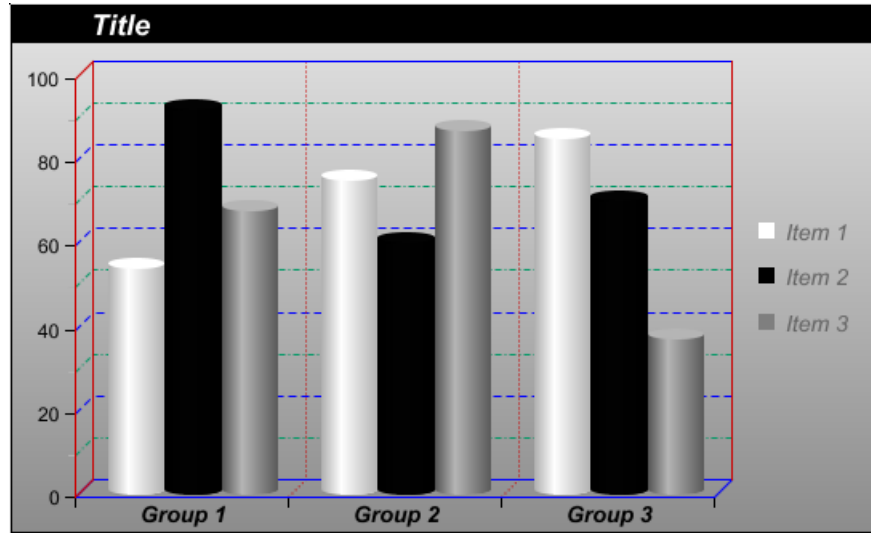


Properties and attributes related to grid line color are available from the `Grid` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- SCALES AND GRIDS
- *Grids*
-
-

GRID LINE STYLES

Corda Builder supports changing line styles for a scale's major grid lines and minor grid lines. Line style options include **plain**, **dashed**, **dotted**, and **dash-dot**.

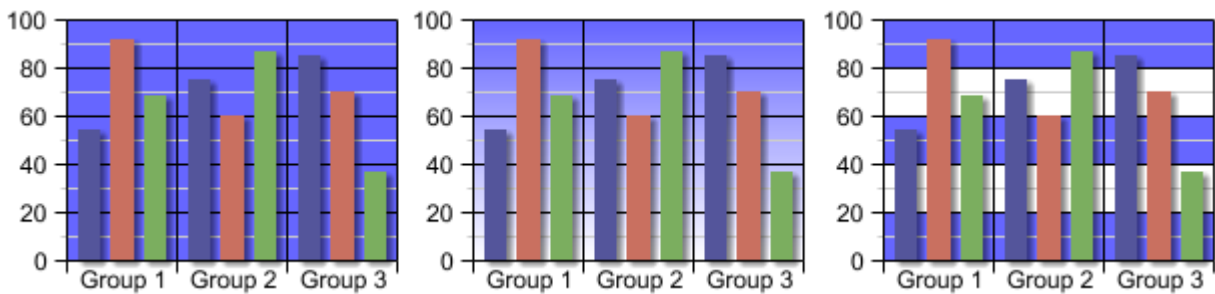


Properties and attributes related to grid line style are available from the `Grid` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

BACKGROUND COLORS

Background colors apply to the area that appears behind a graph's data items. A graph's background is divided into three components: **Back**, **Side** (3D graphs only), and **Bottom** (3D graphs only). A graph's background does not include scale or tick mark areas.

Background colors can be solid fill, gradient fill, or grid stripes, which display alternating colors between major grid lines. Grid stripes can be used with, or in place of, grid lines.



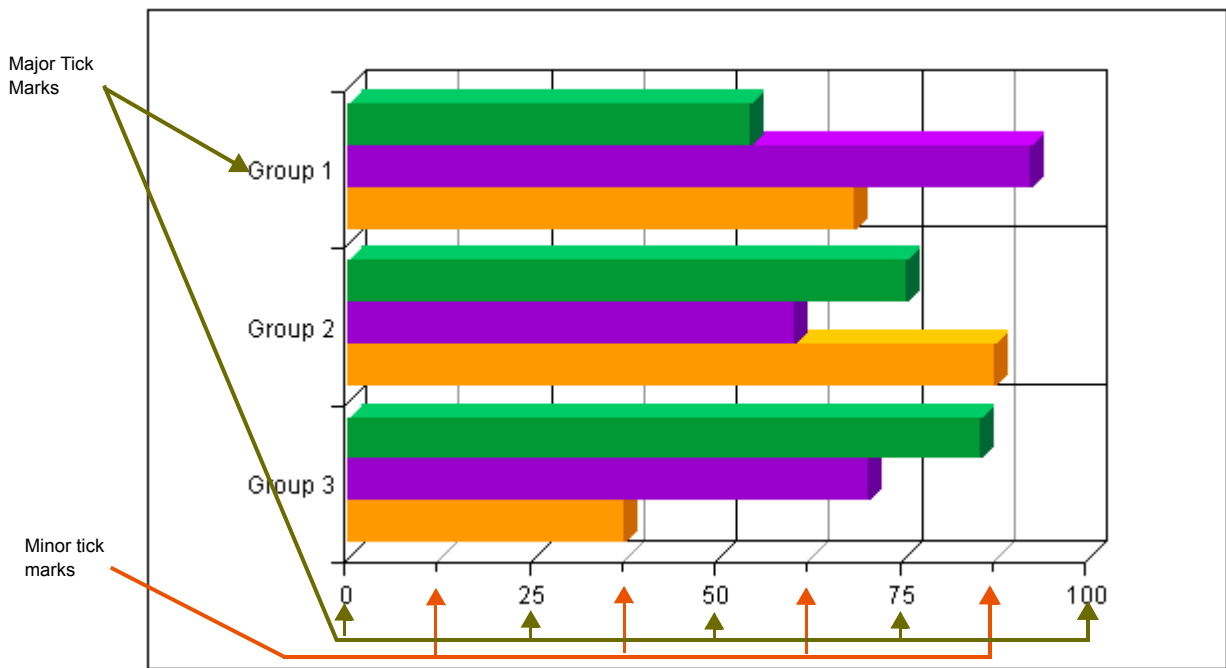
Properties and attributes related to background color are available from the [Grid > Background](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- SCALES AND GRIDS
- Tick Marks
-
-

TICK MARKS

Immediately outside of the graph's grid area are tick marks that are used to indicate specific values along a scale. In most cases, the tick marks are extensions of the grid lines. There are two types of tick marks: *major* and *minor*.

The number of tick marks is regulated by the number of increments in the scale (or the number of categories in category scales). See ["Value Scale Divisions" on page 5-8](#) for details. Tick marks appear at the outer extremes of the scale, as well as between each increment.



Properties and attributes related to tick marks are available from the `Grid > Ticks` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

SCALE MARKERS

Scale markers are accenting lines or shaded background ranges that can be placed on the graph to mark key data points or ranges. The scale marker is not a separate object in the graph like a legend or text box; rather it is an internal object like a bar or line.

Properties and attributes related to a graph's scale markers are available from the [Scale Markers](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

To add a scale marker, select [Insert > Insert Scale Marker](#), select the [Add a Scale Marker](#) button in the toolbar, or click the [Add](#) button next to the Scale Marker property in [Object Properties](#).

To delete a scale marker, select the graph, and click the [Delete](#) button next to the appropriate [Scale Markers > Marker](#) property in [Object Properties](#).

This section discusses the following topics:

- [Scale Marker Configuration](#)
- [Scale Marker Labels](#)
- [Dynamic Scale Markers](#)

Note: *Pie charts, radar graphs, and heat maps do not support scale markers.*

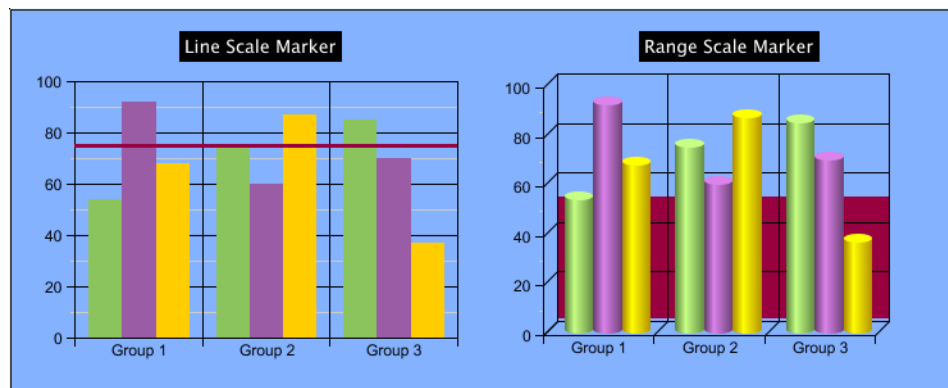
- SCALES AND GRIDS
- Scale Markers
-
-

SCALE MARKER CONFIGURATION

Scale markers come in two types: *Line* and *Range*. In either case, Image template designers can specify the marker color.

LINE MARKERS Line scale markers are single lines that appear in front of data items in the graph. When using a Line marker, specify both the line width and position on the graph.

RANGE MARKERS Range scale markers are colored ranges that occupy the grid area behind the graph data. When using a Range marker, specify both the high and low values for the range area.



Important: *In graphs where a scale marker can appear on multiple scales, specify with which scale the marker should be associated.*

Configure the scale marker using the **Marker** property for the specific scale marker you want to modify. More information on this property is available in the [Corda 7 Object Reference](#).

SCALE MARKER LABELS

Scale marker labels help viewers understand the purpose of the scale marker. Scale marker labels can be displayed directly in the graph, in the graph's legend, or in both.

When displaying the scale marker label, you can specify the label's position, font characteristics, text alignment within the label, border characteristics, and border usage.

Note: *When you choose to display on hover, a small label displaying the letters **RO** (rollover) is displayed inside of **Corda Builder** to identify where the data label appears. These letters do not appear in the final Corda image. When you view the Corda image on the web, the data label only displays when you mouse over a scale marker in the graph.*

Configure the scale marker label using the **Marker > Label** property for the specific scale marker you want to modify. More information on this property is available in the [Corda 7 Object Reference](#).

DYNAMIC SCALE MARKERS

Using an Image Template file with Corda Server lets you add scale markers dynamically based on the live data. These dynamic scale markers override those defined in Corda Builder. Enable dynamic scale markers with ITXML or PCScript.

ITXML

To add a dynamic scale marker to a graph with ITXML, add a **ScaleMarker** property to the desired graph. Adding a scale marker in ITXML must be done when sending the data to Corda Server. More information about this property is available in the *Corda 7 Object Reference* (see "[Corda 7 Object Reference](#)" on page 2-18).

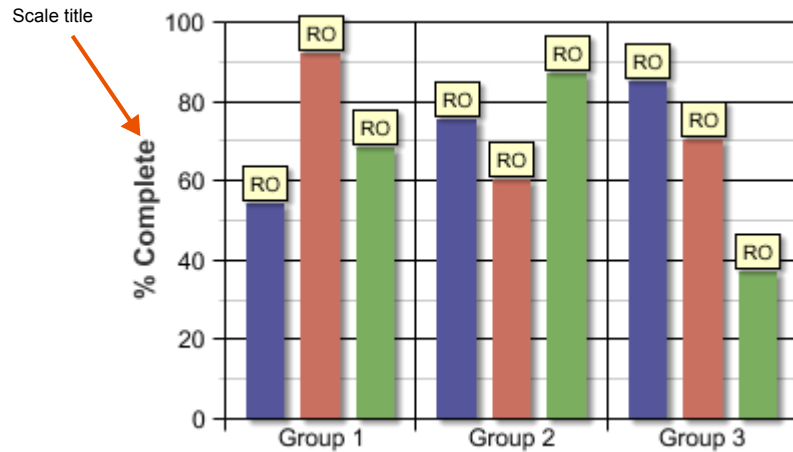
PCSCRIPT

To add a dynamic scale marker with PCScript, use the `graph.AddScaleMarker()` method.

- SCALES AND GRIDS
- *Scale Titles*
-
-

SCALE TITLES

Scale titles allow you to easily create a scale title along the side of the graph, rather than creating, rotating, and positioning a separate text box to serve as the scale title.



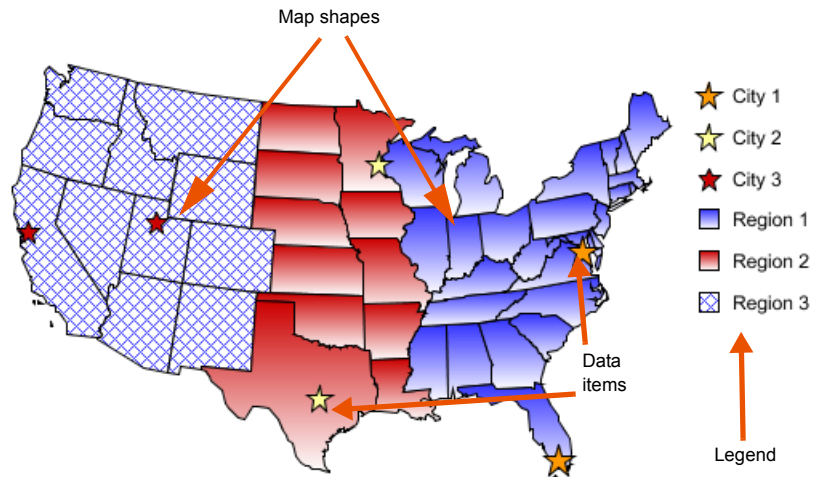
Scale titles appear on the same side of the graph as the scale (either the left or right side).

Properties and attributes related to scale titles are available from the [Value Scale > Title](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

MAPS

This chapter introduces map objects and explains what maps are, how they represent data, and how they can be created and customized.

Maps are most often used to visualize data related to specific locations, whether geographical or otherwise. Data items that compose the map behave based upon the values they represent, providing a visual representation of the data within its location context. Maps can be nested to drill down from general data representations to more specific data representations.



Topics in this chapter include the following:

- [About Maps](#)
- [Map Data](#)
- [Customizing Maps](#)

- MAPS
- *About Maps*
-
-

ABOUT MAPS

Before working with maps, it is important to understand how they work. This section introduces the following topics:

- [Map Layers](#)
- [Maps and Data](#)
- [Map Formatting](#)
- [Map Availability](#)

MAP LAYERS

The building blocks of every map are *map shapes*. Shapes in maps are referred to in this way to distinguish them from regular shape objects (see [“Other Corda Objects” on page 8-1](#)). Map shapes can be grouped together in different ways to create a map. These shape collections are then layered together to form complete maps. Hence, these building blocks are also known as map *layers*. Each layer has its own data and formatting settings.

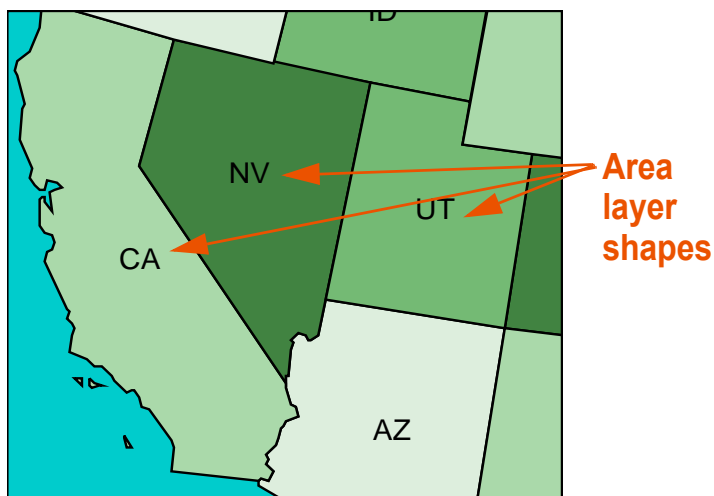
Maps can contain as many layers as necessary to accomplish their purpose. The following layer types are discussed below:

- [Area](#)
- [Points](#)
- [Line](#)
- [Decoration](#)
- [Regions](#)

AREA

The *Area* layer is the foundation of a map. It is typically a collection of map shapes that forms the map background. For example, a map of the U.S. states starts with an Area layer of 50 shapes, each representing a state. Area layer shapes seldom change form or shape themselves because the shapes are critical to the meaning of the map. However, the shape's color and other aspects can change, depending on the value of its data.

Example 6.1 Map Shapes



- MAPS
- About Maps

POINTS

The *Points* layer overlays the *Area* layer to provide more specific data values on a map. Because it does not typically provide actual map structure, the *Points* layer is more flexible in representing data, including the ability to modify shape. The size, shape, and color of *Points* layer shapes can vary according to the data. For example, a *Points* layer might graph data for cities or indicate where certain events took place.

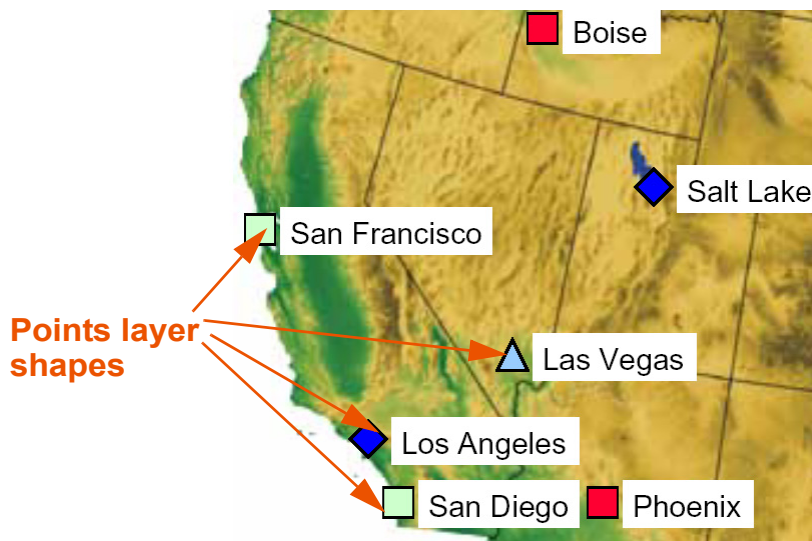
Example 6.2

Area and Points Map Layers



Note: *Area and Points map shapes do not necessarily have to represent data values. They may be used for geographical reference only. For example, a map where the data applies only to the Area layer might still display a Points layer without associated data, in order to provide additional data context, and vice versa.*

Alternatively, a Points layer can be overlaid on a Area layer composed of a bitmap image. For example, a topographical image of the United States serves as the background below.

Example 6.3**Points Layer with Background Image****LINE**

The *Line* layer is used, as its name indicates, to provide line shapes for use on a map. Lines can be used for roads, routes, or other such data items. Color, thickness, and type of line (solid, dashed, etc.) can all be modified based on the data value associated with the line shape.

DECORATION

The *Decoration* layer is not used to represent data values. Its sole purpose is to provide additional detail or context to the map. For example, lake and river shapes might be part of the decoration layer on a geographical map.

REGIONS

A map region is a special case of map layer that allows you to create overlays from existing Area map layers for the purpose of representing additional information. For more information on map regions, see [“Creating a Map Region Layer”](#) on page 6-18.

- MAPS
- *About Maps*
-
-

MAPS AND DATA

In most cases, the color of an Area or Points layer map shape changes depending on the range of its data value. For example, in [Example 6.2](#) above, the state (map shape) of *Arizona* has a value of 21, which puts it in the lightest-colored range, and the state (map shape) of *Nevada* has a value of 98, which puts it in the darkest-colored range. Additionally, Points map shapes may change depending on the data value. For example, the city of *San Francisco* shape has a value of 10, which puts it in the triangle range, while the city of *Las Vegas* shape has a value of 20, which puts it in the square range. Range behavior is defined separately for each map layer (see [Chapter 7](#)).

A map stores its data in a separate table for each layer. Each layer table contains a name for each data item and a corresponding data value. In this way, maps act differently than graphs because the number of data items is predetermined. New map data items must be defined explicitly in Corda Builder. [Example 6.4](#) illustrates how data from these tables is translated into the viewable map.

Example 6.4

How Map Data Works

Value ranges:

Range	Min	Max	Color
< 0	Min	0.0	Yellow
0 - 25	0.0	25.0	Light Green
25 - 50	25.0	50.0	Medium Green
50 - 75	50.0	75.0	Dark Green
75 - 100	75.0	100.0	Very Dark Green
> 100	100.0	Max	Red

CA	45
NV	98
UT	64
ID	56
AZ	21

The state of *Nevada* (in the Area layer) has a value of 98. This value is in the 75-100 range. Therefore, it is colored dark green.



The city of *Phoenix* (in the Points layer) has a value of 30. This value is in the 30-39 range. All Points layer data items in this range are colored yellow and use a diamond symbol.

Los Angeles	20
San Francisco	10
Salt Lake City	0
Phoenix	30
Las Vegas	20

Value ranges:

Range	Min	Max	Color
No name	0.0	9.0	Circle
Medium	10.0	19.0	Triangle
Large	20.0	29.0	Rectangle
Huge	30.0	39.0	Diamond

- MAPS
- *About Maps*
-
-

MAP FORMATTING

“[Maps and Data](#)” on page 6-6 describes how the color of data items can vary according to the range of their data value. The actual shape of a Points layer data item can also vary according to its range. However, color and shape are not the only properties that can vary according to range. Just about any property, including fill, label, hover text, drilldown, and descriptive text can also vary according to range.

The way a map applies its different types of formatting is an important aspect of map behavior. Each layer in a map has three levels of formatting that determine the look and feel of the layer’s data items:

- [Default Layer Properties](#)
- [Range Overrides](#)
- [Shape Overrides](#)

DEFAULT LAYER PROPERTIES

Each layer that composes a map (Area, Points, Line, and Decoration) has a set of default properties. In the first level of formatting, each shape in a given layer receives a look and feel based on these default properties.

If a data item in the layer doesn’t fall in a range and doesn’t have any individual overrides, it looks exactly like the default shape.

Properties and attributes related to default layer properties are available from the [Default Shape](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

RANGE OVERRIDES

After formatting each data item according to the default layer properties, a map checks to see if the data item falls under a range override defined for that layer. For example, in [Example 6.4](#), the map looks at the value for *California* and realizes that it falls in the *25-50* range.

If a data item does fall within a range, it looks to see if the range overrides any of the data item’s default settings. Range overrides can override color or fill settings on Area layers, and, in the case of Points layers, the shape definition. They can also be used to override other settings, such as drilldown or hover shapes and text.

Properties and attributes related to range overrides are available from the [Ranges](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

SHAPE OVERRIDES

Finally, a layer checks to see if the individual data item has any overrides. Typically, this answer is *no* because it is usually a bad practice to override settings for individual data items. An override decreases the flexibility of a project and makes it more difficult to update in the future. When using it as an Image Template file for Corda Server™, it may render the map unsuitable to display certain data sets.

However, there are some instances when it makes sense to override individual data item settings. For example, you might want to add hover items to an Area layer data item. Or the label on a certain shape might inconveniently overlap with the label of another shape, making it necessary to change the position of one of the labels.

Properties and attributes related to shape overrides are available from the `Map Shape` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

Note: *Because maps are just collections of shape objects, any characteristic of a shape can be overridden for a particular map shape as well.*

MAP AVAILABILITY

Corda 7 offers a wide variety of world, country, U.S., state, and county maps. Some of these maps are included with the purchase of Corda Builder or OptiMap™. Other maps must be purchased separately.

To keep the download size of Corda 7 small, the Corda 7 installer includes a small number of sample maps. Licensed Corda 7 users are entitled to a number of other maps, which can be downloaded from <http://www.corda.com/download/maps>.

For a catalog of available maps, see the *Corda 7 Map Guide*.

- MAPS
- Creating Maps
-
-

CREATING MAPS

There are three ways to add a map to a project:

- [Importing a Predefined Map](#)
- [Creating a New Map from Shapes](#)
- [Creating a New Map from an Image](#)

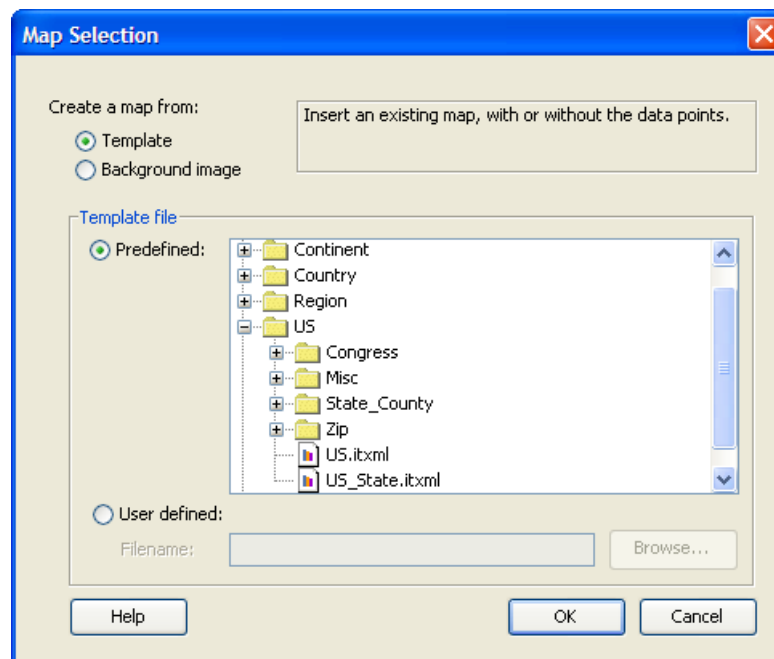
IMPORTING A PREDEFINED MAP

When adding a map, select a template upon which to base the map. Many templates are available, including maps for countries, U.S. states, and other geographic regions. For more information, see the *Corda 7 Map Guide*.

Map templates might already have default formatting settings, which can be further customized after adding the map to a project.

To add a map

- 1 Select **Edit > Create Map** or select the **Create Map** icon in the toolbar



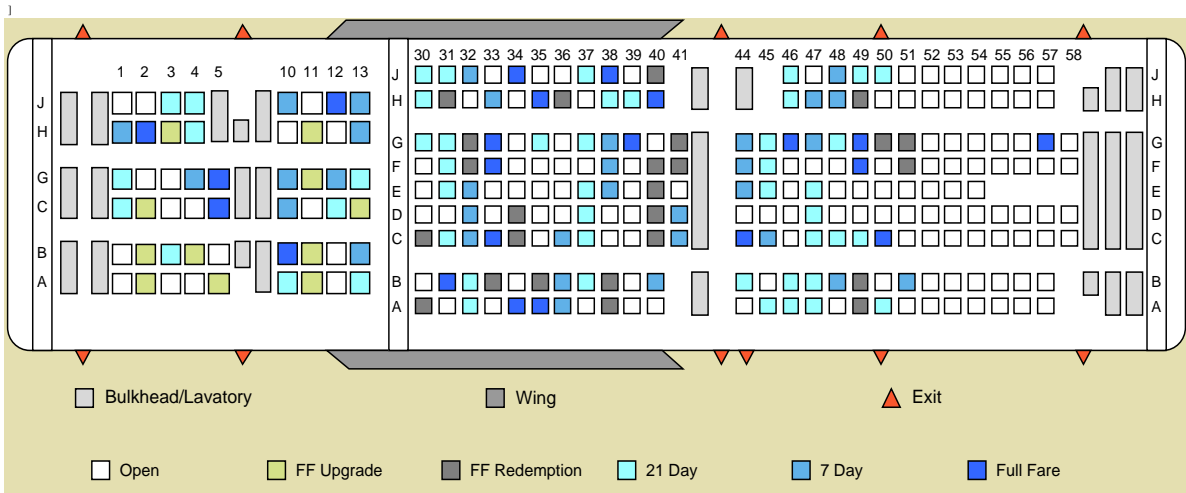
2 Select a map template.

Select a map template from the template tree in the center of the dialog, or select **User Defined** and browse for a different template.

3 When you have selected the desired template, select **OK or **Apply** to finish adding the map to the project.**

CREATING A NEW MAP FROM SHAPES

Creating a new map out of regular shape objects is not the best way to create a map when the boundaries to the shapes are complicated (such as in a state map of the United States), but it can be a great way to create maps where the shapes are Points, such as in a desk layout map of an office or in a seating chart for an airplane.



To create a map from shapes

1 Create and arrange the shapes of which the map will be composed.

To learn more about creating shape objects, see [“Shapes” on page 8-6](#).

For information about disbanding a map and using/rearranging the shapes within it, see [“To disband a map” on page 6-16](#).

2 Select all of these shapes at once.

If you do not know how to select multiple shapes at once, see [“Selecting Objects” on page 3-2](#).

3 Select **Map > Combine into New Map.**

- MAPS
- Creating Maps
-
-

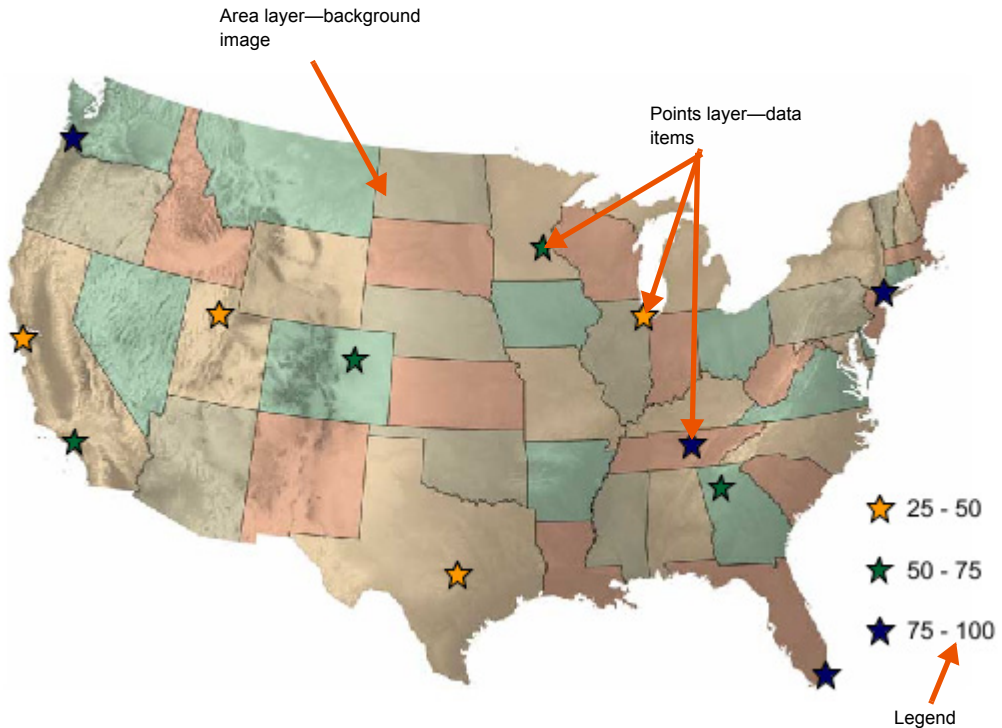
Or right-click and select **Combine into New Map**.

- 4 Enter the map name in the **Create New Map** dialog that appears.
- 5 Click **OK**.

You now have a new map in the project, composed of the shapes you created.

CREATING A NEW MAP FROM AN IMAGE

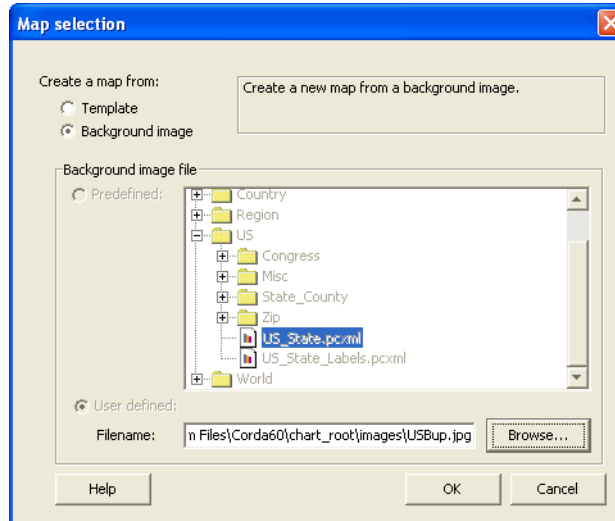
A map can use a background image in place of shapes, with point data items plotted relative to the background image.



Note: If you base a map on a bitmap image, the image replaces the Area layer, leaving the Points layer to represent the data items you may have to overlay on the image.

To create a map from a background image

- 1 Select **Edit > Create Map**, or select the **Create Map** icon in the toolbar



- 2 Select **Background Image**.
- 3 Click **Browse** to locate a background image.
- 4 When you have selected the desired image, click **OK** to finish adding the map to the project.

- MAPS
- Map Data
-
-

MAP DATA

Map data can be entered all at once, or it can be applied to each map shape individually. The following steps explain how to do this.

- [Map Layer Data](#)
- [Map Shape Data](#)

After entering map data, it might be necessary to edit the map's data ranges, which define rules for how Corda Builder should depict the map data. For more information, see Chapter 9, "[Map Ranges.](#)" To review how Corda maps represent data, see "[Maps and Data](#)" on page 6-6.

MAP LAYER DATA

Corda Builder stores map data in multiple tables—one for each layer used in the map. Each row in the data table represents a separate data item. There are four columns in each table:

- *Name*: The first column represents the name of a data item. This name must correspond to a data item that has been defined within the selected map layer. (In this behavior, maps act differently than graphs—the number of data items is predetermined.)
- *Long Name*: The second column represents the long name for the data item.
- *Code*: The third column represents the data item's code name.
- *Value*: The fourth column represents the value of the data item. This is the only editable column.

To edit map layer data

- 1 Select a map layer in the **Object List**.
- 2 Open the **Data Editor** in **Object Properties**.

Click the Ellipsis [...] button to open the **Data Editor**.

(* sorted column)

Name *	Long name	Code	Value
Abilene, TX			No data
Albany, GA			No data
Albany, NY			No data
Albuquerque, NM			No data
Alexandria, LA			No data
Allentown, PA			No data
Altoona, PA			No data
Amarillo, TX			No data
Anchorage, AK			No data
Annapolis, MD			No data
Anniston, AL			No data
Appleton, WI			No data

Help OK Cancel

- 3 Modify the data in the **Value** column, as needed.

The **Value** column contains the data values for each data item in the selected map layer.

Enter data manually into the data table by selecting a cell and entering a number, or paste data from outside sources, including spreadsheet applications such as Microsoft Excel. All of the usual cut, copy, and paste shortcut keys (**Ctrl+x**, **Ctrl+c**, **Ctrl+v**) work in the **Data Editor**.

- 4 Click **OK** to save the changes.

MAP SHAPE DATA

To change the data for an individual shape in a map layer, select the appropriate object in the **Object List**, and edit the shape value in **Object Properties**.

Properties and attributes related to map shape data are available from the **Shape Settings** property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- MAPS
- Customizing Maps
-
-

CUSTOMIZING MAPS

You need to customize maps to their specific needs once they have been created. This section discusses the following customizations:

- [Working with Layers](#)
- [Working with Individual Map Shapes](#)
- [Hiding Shapes with No Data](#)
- [Other Data Item Formatting](#)

As discussed in “[Map Formatting](#)” on [page 6-8](#), maps can be customized on three different levels. This section discusses both map-wide customizations, and data-item-specific formatting. Additional information on customizing a map through the use of map ranges is available in [Chapter 7](#), *Map Ranges*.

Move, resize, and delete the entire map as you do any other object in Corda Builder (see “[Common Object Actions](#)” on [page 3-2](#)). In addition, maps can be disbanded back to their component shapes.

To disband a map

Instead of deleting a map, Corda Builder lets you disband it, leaving behind its shapes. This is useful for using shapes from one map to create a new map.

- 1 **Select the entire map.**
- 2 **Select [Map](#) > [Convert Map to Shapes](#).**

The map disappears, and the map shapes are converted to regular shape objects. Each resulting shape is listed separately in the [Object List](#).

WORKING WITH LAYERS

As discussed in “[Map Layers](#)” on [page 6-2](#), maps consist of one or more layers. The two most common layers are Area and Points. The properties for these layers (data, colors, fill settings, outlines, drilldown, etc.) are independent of one another. Thus, when customizing a map, make sure to select the correct layer to customize. For information on modifying map data, see “[Map Data](#)” on [page 6-14](#).

Because, at their most basic, map layers are collections of shapes, there are some options available for customizing the look and display of layers and shapes within a map:

- [Dividing a Map Shape](#)
- [Creating a Map Region](#)
- [Hiding Shapes with No Data](#)

DIVIDING A MAP SHAPE

When dividing an individual map shape into multiple shapes, the process is the same as the process used to divide a regular shape object. You create a divider object and then divide the map shape along the boundary created by the divider object. For more information on dividing a shape, see [“Dividing a Shape” on page 8-19](#).

Note: *To manipulate an individual map shape, use the [Select a Point](#) cursor from [Shape Tools](#). For more information, see [“Select a Point” on page 8-13](#).*

CREATING A MAP REGION

Creating a map region allows you to take existing map shapes and combine them together into a new shape. For example, you might want to organize a map of the United States into five regions, instead of labeling every state.

To create a map region

- 1 **Add a base map to the project.**

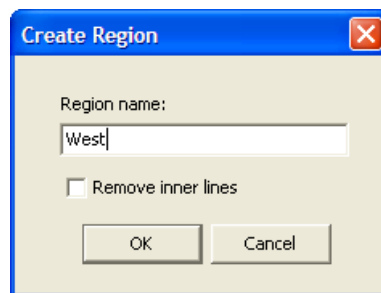
For more information on creating maps, see [“Creating Maps” on page 6-10](#).

- 2 **Select the existing map shapes for which you want to create a new map region.**

Do this by pressing the <Ctrl> key and selecting each individual map shape, either from the [Object List](#) or in the project window.

Note: *To select individual map shapes in the project window, you must enable [Edit > Preferences > Advanced map edit mode](#).*

- 3 **Select [Map > Combine into new region](#).**



Select [Remove inner lines](#) if you want to eliminate the boundary lines between the original shapes when the new region shape is created.

- 4 **Click [OK](#) to create the new region shape.**

- MAPS
- Customizing Maps
-
-

CREATING A MAP REGION LAYER

A variation on map regions makes a copy of the map's existing Area layer and uses it to define new region shapes. This has the advantage of maintaining the previously existing shape data as you create the new region shapes.

For example, you might want to break up an existing map of the United States into sales regions, each containing multiple-state map shapes. To preserve the label, fill, hover, and other data associated with each state object, use a region layer to represent the sales regions and overlay that layer on the existing map in order to preserve existing data and still represent the new region information.

To create a region layer

- 1 **Add a base map to the project.**

For more information on creating maps, see [“Creating Maps” on page 6-10](#).

- 2 **In the [Object List](#) or [Project Canvas](#), select the map to which you want to add a new region layer.**

- 3 **Select [Properties](#) > [Copy Area Layer](#).**

- 4 **From the dropdown list, select the appropriate [Area Layer](#) to copy.**

To distinguish it from the original Area layer, the new region layer has thicker boundary lines between the shapes that comprise the layer.

After it is created, the new region layer can be manipulated just as any other Area map layer, including the creation of new region shapes. For more information on creating map regions, see [“Creating a Map Region” on page 6-17](#).

HIDING SHAPES WITH NO DATA

In creating the map, you might decide to not use all of the data items defined in a map. Instead of deleting them, the map can hide data items with no data in the published image. Data items without associated data still appear on the [Project Canvas](#), but when you preview or publish the map, empty data items are excluded.

Note: *Although this option is available for the Area layer, it doesn't usually make sense to use it because data items in this layer typically contribute to map structure even if there is no data associated with a given data item. Regardless of the layer in question, data items must have a value of [No Data](#) (the default setting) to be hidden.*

Properties and attributes related to hiding shapes with no data are available from the [Layer Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

WORKING WITH INDIVIDUAL MAP SHAPES

As discussed in [“Map Formatting” on page 6-8](#), the most granular point at which to apply formatting is the individual data item (map shape). Map shape properties are typically controlled as a group through layer-level formatting and data ranges, but each map shape maintains its own formatting properties and can be managed individually, if necessary. For information on modifying map data, see [“Map Data” on page 6-14](#).

SELECTING MAP SHAPES

Select individual map shapes from the [Object List](#) on the left side of the project in Corda Builder. Once selected, modify the map shape’s properties in [Object Properties](#).

Properties and attributes related to individual map shapes are available from the [Shape Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

ADDING POINTS LAYER DATA ITEMS

Most map templates include a Points layer with some pre-defined data items. However, you might find that you need to add data items to the map’s Points layer. This is an easy task to perform.

To add a Points layer data item

- 1 **Right-click the location on the map where you want to add the data point and select [Add a data point to map](#).**

Alternatively, select [Properties > Add a data point to map](#), or hold down the <Alt> key as you click a point inside the map.

- 2 **Enter the data item name, value, and description in the following dialog:**

If you have latitude and longitude coordinates for the data item, enter them here as well.

- 3 **Select [OK](#) to finish adding a data item to the map.**

- MAPS
- Customizing Maps
-
-

ADDING AREA DATA ITEMS

To add new data items to the map's Area layer, first create it by either using the **Edit > Create Shape** menu option, or by disbanding another map and selecting one of its former shapes (see ["To disband a map" on page 6-16](#)).

To add a Area layer data item

- 1 **Select both the map and the shape to add to the map.**

Hold down the <Ctrl> key to select multiple objects at once.

- 2 **Select **Properties > Add Shape to Map Layer**.**

From the dropdown list, select the appropriate Area layer to which you want to add the shape.

DELETING DATA ITEMS

Although you can delete unnecessary data items, it might be better to hide data items that don't contain data. For more information, see ["Hiding Shapes with No Data" on page 6-18](#).

To delete a data item

- 1 **Select the data item you want to delete.**
- 2 **Select **Edit > Delete Object** or press the Delete key.**

Corda Builder lets you delete unnecessary data items, but hiding map data items without data might be a better alternative to deleting the data items completely. For more information, see ["Hiding Shapes with No Data" on page 6-18](#).

MOVING DATA ITEMS

Move map data items around in the map if the default placement isn't effective.

Note: *When moving a data item on the map, make sure to move it relative to the map, not the Project Canvas.*

To move a Points layer data item

- 1 **Select a Points layer data item.**

Be careful to select a specific data item instead of the entire map. The data item is outlined when moused over, indicating that it can be selected.

- 2 **With the data item outlined, click and drag it to the desired location.**

Points layer data items don't retain original position data, so you can't restore a Points layer data item to its original location.

To move an Area layer data item

1 Open Shape Tools and select the **Select a Point** cursor.

Remember that map data items are just shapes. For more information, see [“Select a Point” on page 8-13](#).

2 Select the Area layer data item.

The **Select a Point** cursor prevents you from selecting the entire map.

3 With the data item outlined, click and drag it to the desired location.

To return a background shape to its original position, select the shape in the **Object List**, right-click it, and select **Restore to Original Position**.

OTHER DATA ITEM FORMATTING

Within a data item’s properties are properties for configuring general settings such as symbol type (Points layer only), fill, and outline, and other related formatting such as labels, hover, drilldown, and descriptive text. However, it is important to remember that a published map’s final settings are typically defined by the range properties defined for each map layer. Default properties are always overridden by range properties (unless a data item has no data), while range properties are overridden by individual map shape settings.

Important: *To always use a map layer’s default settings, uncheck **Layer Settings > Use Ranges in Object Properties**.*

Warning: It is generally a bad practice to override formatting for an individual data item. If the project is used as an **Image Template** file with **Corda Server**, it might make the map impractical to use with certain data sets, and less flexible to a developer who might want to change its formatting on-the-fly.







Many data item formatting options are intuitive, but consider the following to use some of the formatting options effectively:

- [Symbol Notes](#)
- [Fill Notes](#)
- [Label Notes](#)
- [Drilldown Notes](#)
- [Hover Notes](#)

- MAPS
- Customizing Maps
-
-

SYMBOL NOTES

Corda Builder provides several shape types to use as symbols for Points layer data items:

Shape Types	Shape
Rectangle	
Round	
Triangle	
Diamond	
Rounded Rectangle	
Polygon	

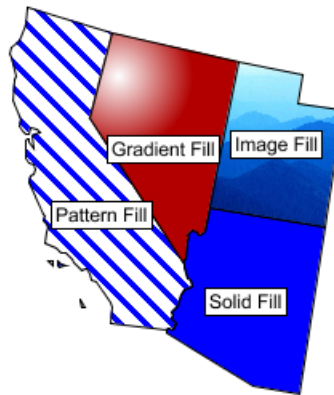
Properties and attributes related to individual map shapes are available from the [Shape Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

FILL NOTES

Corda Builder supports four different types of fill. Specify the type you want to use in the fill-type property:

- **Solid**: This option is the fill color only.
- **Pattern**: This option allows you to select from several pattern fills in the [pattern-type](#) property.
- **Gradient**: This option allows you to adjust the gradient type, colors, and offset values.

- **Image:** This option allows you to browse for an image to add to the map.



Properties and attributes related to a data item's fill are available from the [Shape Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

LABEL NOTES

For more information on labels, see ["Map Data Labels" on page 10-7](#).

One common issue with maps is label positioning. Since many data items can be grouped close together, labels can easily overlap or become cluttered and illegible. To resolve this problem, override one or more of the following label characteristics:

Repositioning or reformatting a label: For more information on repositioning or reformatting a label, see ["Map Data Label Layout" on page 10-8](#).

Overriding a label font: This is generally discouraged since it makes the map less flexible. However, it can be useful for making a label more readable.

Overriding label text: It is generally a bad practice to override the label text for an individual map data item because it makes the project harder to update in the future. If the project is used as an Image Template file with [Corda Server](#), this might also make the map less flexible to a developer who wants to change label text on-the-fly. It is better to define the label text on a global level or at least on a range level.

Properties and attributes related to data item labels are available from the [Label](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- MAPS
- Customizing Maps
-
-

DRILLDOWN NOTES

For information on drilldown, see [“Map Drilldown” on page 9-7](#).

Properties and attributes related to data item drilldown are available from the `Drilldown` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

HOVER NOTES

A powerful feature of Corda Builder is the ability to create *hover items*—that is, maps, graphs, shapes, and other objects that “pop up” as you mouse over data items in the map.

Note: *Hover shapes work only in Flash* and SVG output.*

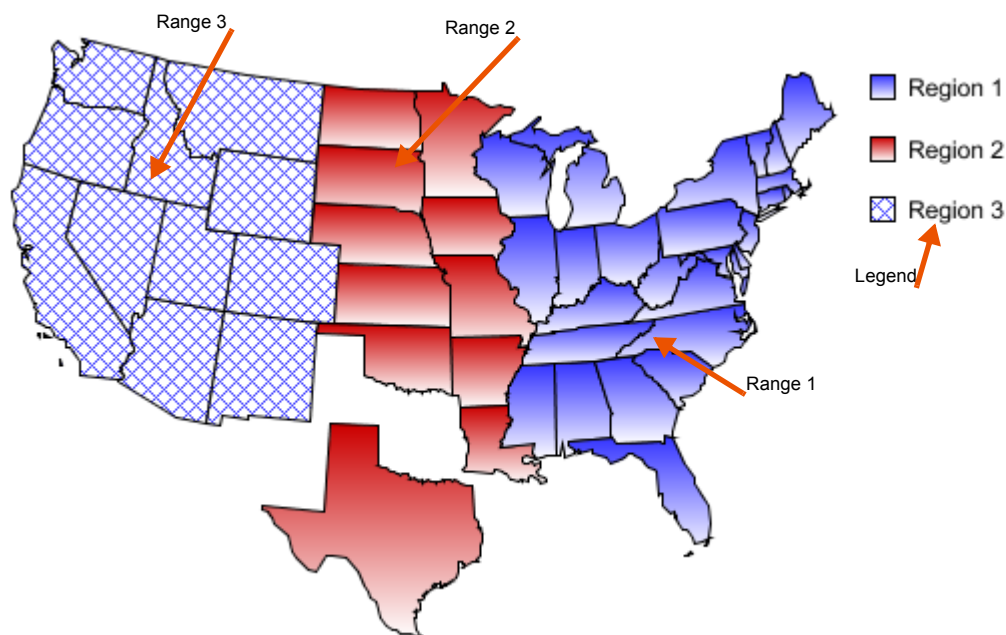
To do this, you first need to define another object in the map as a hover item (see [“Defining a Hover Object” on page 11-2](#)). To make the object “pop up,” define a trigger shape, as explained in [“Defining a Trigger Shape” on page 11-2](#).

For information about hover text in maps, see [“Map Hover Text” on page 10-13](#).

Properties and attributes related to data item hover text are available from the `HoverText` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

MAP RANGES

Map ranges determine the “look and feel” of the data items in a given map layer, based on the value assigned to that data item. For example, three ranges in the map below determine the formatting for the state shapes that comprise the map of the United States.



Each range can define its own formatting, label, drilldown, and hover settings. Any data item whose value falls in that range inherits the range settings and uses them instead of the default shape properties defined for the data item in this map layer. Usually, you want to distinguish data items that fall in a certain range by changing at least the color (and possibly the symbol type for data items) of that range.

This chapter describes how to add ranges to the map and how to customize the format of the ranges in the map. This chapter consists of the following topics:

- [Range Generation](#)
- [Range Calculation](#)
- [Range Names](#)
- [Range Settings](#)

- MAP RANGES
- Range Generation
-
-

RANGE GENERATION

Corda Builder can automatically generate ranges, or let you define them manually:

- [Automatic Range Generation](#)
- [Manual Range Generation](#)

AUTOMATIC RANGE GENERATION

Corda Builder can automatically generate a specified number of ranges and customize their color (and symbol for Point layer data items) settings. To do this, provide the following information:

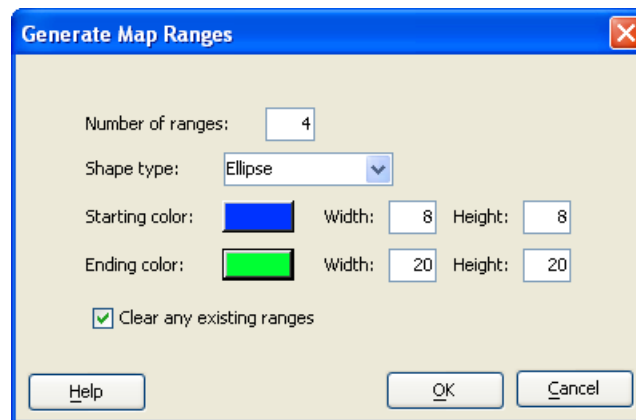
Area layer ranges: Specify the number of ranges to generate, a starting color, and an ending color. The first range is assigned to the starting color. The last range is assigned to the ending color. The remaining ranges are assigned gradient colors between the starting and ending colors

Point layer ranges: Specify the number of ranges to generate, a symbol type, starting color and size, and an ending color and size. Each range uses the specified symbol type. The first range is assigned to the starting color and symbol size; the last range is assigned to the ending color and symbol size. The remaining ranges are assigned gradient colors between the starting and ending colors. The symbol size progressively shifts from the starting size to the ending size.

Corda Builder then adds the specified number of ranges to the range list.

To have Corda Builder generate ranges

- 1 **Select the map.**
- 2 **Select the layer to edit in the **Object List**.**
To learn more about map layers, see ["Working with Layers"](#) on page 6-16.
- 3 **Open the **Ranges** property and click the **Ellipsis [...]** button in the **Generate** field.**
The **Generate Map Ranges** dialog appears.



- 4 **Specify the number of ranges to generate in **Number of ranges**.**
- 5 **(Conditional) For Point layer data items, select the symbol type from the **Shape type** drop-down list.**
- 6 **Click **Starting color** and **Ending color** to set the starting and ending colors for the ranges.**

The first range is assigned to the starting color. The last range is assigned to the ending color. The remaining ranges are assigned to gradient colors between the starting and ending colors.

- 7 **(Conditional) For Point layer data items, specify the starting and ending symbol sizes in the **Width** and **Height** fields.**

The first range is assigned to the starting symbol size. The last range is assigned to the ending symbol size. The symbol sizes for other ranges grows progressively between these values.

- 8 **Select **OK** to save.**



This adds the specified number of range properties to the selected map layer properties.

- MAP RANGES
- *Range Generation*
-
-

MANUAL RANGE GENERATION

Sometimes it is necessary to add or delete a range in a map layer.

To add a single range

- 1 **Select the map.**
- 2 **Select the layer to edit in the **Object List**.**
To learn more about map layers, see ["Working with Layers" on page 6-16](#).
- 3 **In **Object Properties**, click the **Add** button  in the **Ranges** field.**
A new *Range* property is inserted at the beginning of the map layer's range list. Edit the new range's properties and attributes as needed.
- 4 **To delete a range, simply click the **Delete** button  next to the appropriate Range name in **Object Properties**.**

RANGE CALCULATION

An important aspect of a map is how it determines which data items belong to each range. This is a process known as range calculation, and is mostly invisible for end users because of the automatic calculation abilities of Corda 7. However, advanced users have many options for controlling how maps calculate their ranges.

Each range has minimum and maximum values. Any data item whose value falls between the minimum and maximum values, inclusive, for a range belongs to that range. For example, a range with a minimum of 20 and a maximum of 25 includes map data items with values of 23 or 25, but not data items with values of 19 or 26.

Corda Builder provides three methods for range calculation: two automatic algorithms in which Corda Builder looks at the map data and automatically determines range minimums and maximums, and one manual method that lets map designers specify range minimums and maximums.

To specify a range calculation method

- 1 **Select the map.**
- 2 **Select the layer to edit in the Object List.**
To learn more about map layers, see ["Working with Layers" on page 6-16](#).
- 3 **In Object Properties, open the Ranges property.**
- 4 **Select the desired range calculation method from the Calculation drop-down menu.**
The three options are
 - [\(Auto\) Use Range Percentages](#) (Default)
 - [\(Auto\) Use Data Values](#)
 - [\(Manual\) Use Range Values](#)
- 5 **(Optional) Modify specific range calculation attributes in the map Range properties, as needed.**

(AUTO) USE RANGE PERCENTAGES

By default, a map distributes its ranges according to percentiles. Each range is assigned a certain percent of the data items in the map.

For example, the first range might contain the lowest 25 percent of the data items, and the last range might contain the highest 10 percent of the data items. The actual minimum and maximum values of the ranges change as the data changes, but about 25 percent of the data items are in the first range, 10 percent are in the last range, and so on.

In addition to setting range percentages, maps can automatically compensate for range percentages that result in non-round minimum and maximum values. For example, the third

- MAP RANGES
- *Range Calculation*
-
-

25 percent of data items might fall between 153 and 197. The map notices that by expanding this range to 26 percent of the data items, it can round the range to 150 through 200.

Properties and attributes related to range calculation are available from the [Ranges](#) property, and in the individual [Range](#) properties, in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

(AUTO) USE DATA VALUES

Another automatic range calculation algorithm, Auto Using Values, instructs a map to calculate ranges so that each range is the same size. To do this, the **Corda Builder** examines the map's data items, computes the difference between the largest and smallest data items, and divides this difference by the total number of ranges to define a range size. Minimum and Maximum values are then applied to each range to make them this size. This approach yields evenly incremented ranges.

For example, if a map's data items contain values from 12 to 148 and there are four ranges, each range spans 34 possible values: 12 - 46, 47 - 80, 81 - 114, and 115 - 148.

Note: *You can specify whether or not zero is included in this calculation.*

For many data sets, this approach yields ranges similar to those generated by percentage-based scaling. However, with skewed data sets, such as bell-curves, the ranges are very different. For example, only one or two data items might be in one range, and ninety percent of the data items might be in another range. Thus, this approach is good for highlighting abnormal or outlying data items but might yield poor results if you try to compare closely clustered data.

In addition to setting range sizes, maps can automatically compensate for range sizes that result in non-round minimum and maximum values. For example, the third range might fall between 153 and 197. The map notices that by slightly adjusting the range, it can round the increment to 150 through 200.

Properties and attributes related to automatic range calculation are available from the [Ranges](#) property, and in the individual [Range](#) properties, in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

(MANUAL) USE RANGE VALUES

The manual Use Range Values option lets you define each range's minimum and maximum values manually. Remember, however, that manually calculated ranges do not compensate for unanticipated map data. Because of this, you should only scale a map layer's ranges manually if you are confident that the map data falls into the specified ranges.

Note: *Data item values that do not fall into a defined range are displayed using **Default Shape** properties.*

If you have map ranges set to use-range-values and then delete a range in the middle, the items with values that don't fall inside one of the ranges (i.e., values that used to fall in the range now deleted) will be displayed using the default shape's color (normally white). Items that have no data use the color from the no-data range (it is also white by default). So, by default you cannot tell an item that has no data from one that has data but does not fall in one of the ranges. This is generally an unusual case, but you can make changes to tell them apart.

Important: *Values are evaluated as greater than or equal to the minimum value and less than or equal to the maximum value, with lower values having precedence. For example, if two ranges are defined with minimums and maximums of 0-50 and 50-100, the value 50 is defined to be in the lower range (0-50).*

Properties and attributes related to manual range calculation are available from the individual **Range** properties in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- MAP RANGES
- *Range Names*
-
-

RANGE NAMES

It is often useful to assign names to map layer ranges. This name is used to reference the range in Corda Builder and ITXML. Range names can also be used in hover text, drilldown, labels, and legends.

By default, each range is represented in a map's legend with a label. Manually scaled ranges require you to manually specify each range's label.

Properties and attributes related to range names are available from the individual [Range](#) properties in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

RANGE SETTINGS

A map layer's data item formatting can change based upon the range into which its value falls. Data item formatting includes such things as outline, color, fill type, and symbol type (for Point layer only). For more information on setting a range's formatting options, see ["Other Data Item Formatting" on page 6-21](#).

- **MAP RANGES**
- *Range Settings*
-
-



OTHER CORDA OBJECTS

In addition to the powerful graph and map objects available in **Corda Builder**, other objects are provided that generally play a supporting role to the primary graph or map images in a **Corda Builder** project. These secondary objects can add focus, clarity, and style to the displayed data. This chapter introduces the following objects:

- [Legends](#)
- [Text Boxes](#)
- [Shapes](#)
- [Images](#)

- OTHER CORDA OBJECTS

- *Legends*

LEGENDS

Legends are used to identify ranges, data series, and scale markers within a graph or map. For each range, data series, or scale marker, the legend contains a label. To the left of each label is a small box that indicates the color, symbol type, and image for data items in that range, data series, or scale marker.

Each graph or map can have only one legend. Some map and graph templates include a legend by default, but legends can be added manually to any map or graph.

Add, move, and delete legends within a project as you would any other object in a Corda Builder project. For more information, see [“Working with Objects” on page 3-2](#).

Legend objects have their own unique set of properties. Information on legend properties is available in the [Corda 7 Object Reference](#).

Corda Builder provides several options for configuring legends, including the following:

- [Legend Labels](#)
- [Legend Markers](#)
- [Legend Layout](#)
- [Border and Shadow](#)

LEGEND LABELS

Legend labels are controlled by a legend format string. The format specifies text and/or meta tags to define the label for each data series represented in the legend. However, there are some differences between graphs and maps when it comes to the use of legends.

GRAPH LEGEND LABELS

Graph legends use the `%_SERIES_NAME` meta tag to generate labels automatically from the graph series names. For more information about editing series names, see [“Column and Row Names” on page 4-11](#).

If a scale marker is present, it is also listed in the graph legend. For more information about adding a scale marker, refer to [“Scale Markers” on page 5-27](#).

Graph legend labels support both static text and dynamic meta tags supported by graph objects.

MAP LEGEND LABELS

Map legends use the `%_RANGE_TEXT` meta tag to generate labels automatically from the map range names. For more information about editing range names, see [“Range Names” on page 7-8](#).

Map legend labels support both static text and all dynamic meta tags supported by map objects, but this text only applies to manually scaled ranges. Automatically scaled ranges automatically generate their own legend text values.

Properties and attributes related to legend labels are available from the [Legend Settings > Label](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

LEGEND MARKERS

Legend markers are the colored boxes, symbols, and images that appear to the left of a legend label. Graph legend markers are boxes colored to match the series to which they are associated.

Map legend markers differ depending on the map layer to which they are associated. Point layer legend markers are, by default, the same size, shape, and color as the data items within that range. Area layer legend markers are square and the same color as their associated range.

The size of a range marker is configurable: It can be configured to match the size of the font in the legend label, a specific height and width can be configured for the range marker in pixels, or a maximum range marker size can be configured.

Alternatively, the legend object can manage legend markers automatically. This is useful if range markers are too big compared to the rest of the legend.

Properties and attributes related to legend markers are available from the [Legend Settings > Markers](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

LEGEND LAYOUT

Legends are organized with the legend marker to the left of its corresponding legend label, but [Corda Builder](#) lets you organize a legend's layout as needed. To better control the positioning of legend markers, specify their alignment to a fixed left margin. This is useful when aligning several different legends.

Properties and attributes related to legend layout are available from the [Legend Settings > Layout](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- OTHER CORDA OBJECTS
- *Legends*
-
-

BORDER AND SHADOW

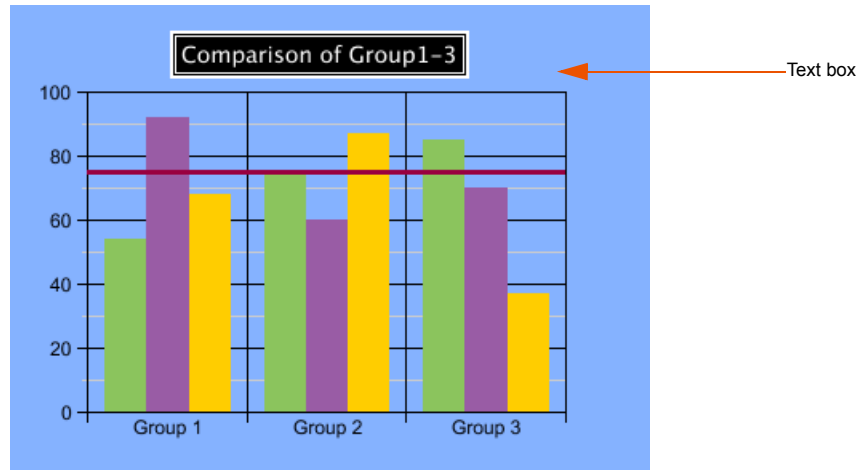
To enhance the look of a legend, add border and/or shadow.

Note: *Shadow options should not be used when the legend is using a transparent background.*

Properties and attributes related to legend border and shadow are available from the `Legend Settings` property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

TEXT BOXES

Text boxes improve the readability of a Corda image by explaining the meaning of the information depicted or by providing titles for objects in the Corda image.



Corda Builder supports Title boxes and other text boxes in Image Template files. Add text boxes using the **Add a Text box** button in the **Tools** view. For more information, see [“Using Shape Tools” on page 8-7](#).

Move and delete text boxes within a project as you would any other object in a Corda Builder project. For more information, see [“Working with Objects” on page 3-2](#).

To define a text box as a title box

- 1 **Select the text box that contains the project title.**
- 2 **Right-click the text box.**
- 3 **Select **Set as Title**.**

This changes the object name to `title`. If another text box was previously set as the title text box, it is given a different name.

Note: *The Title text box is used to generate descriptive text, so it is important to properly create a Title box. For more information on descriptive text, see [“Descriptive Text” on page 11-5](#) and [“Data Tables” on page 11-11](#).*

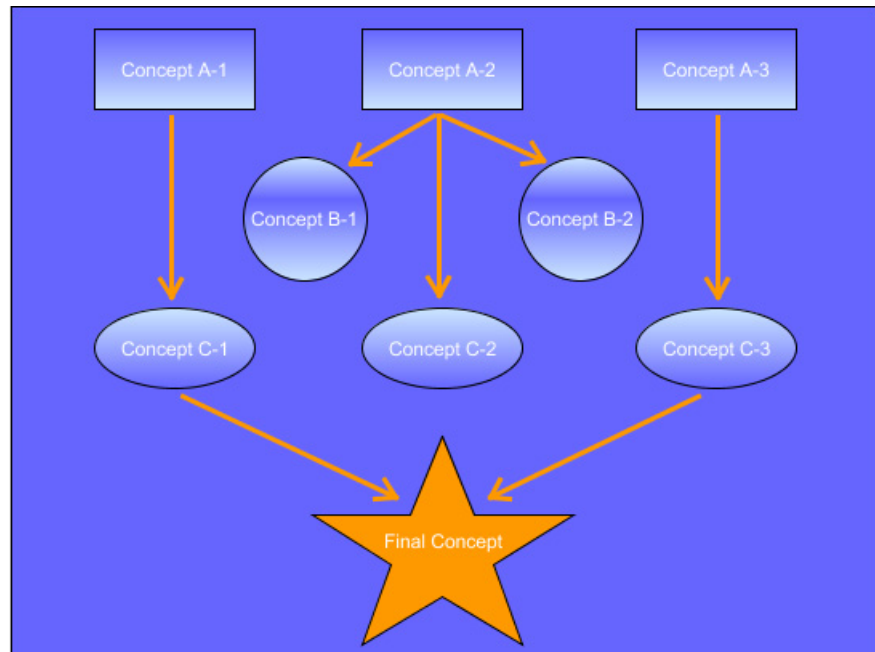
Information on text box properties is available in the [Corda 7 Object Reference](#).

OTHER CORDA OBJECTS

Shapes

SHAPES

Shapes can provide additional focus and organization to a Corda image. They can improve the usability of the graph or map, highlight information that might otherwise be missed, build flow charts, or create callout notes.



Corda Builder provides an advanced set of **Shape Tools** on the left side of the Corda Builder interface for creating, importing, and manipulating shape objects

Note: *Once shapes have been created, move and delete shapes as you would any other object in a Corda Builder project. For more information, see ["Working with Objects"](#) on page 3-2.*

USING SHAPE TOOLS

Corda Builder **Shape Tools** provide robust functionality for creating and manipulating shapes in your projects. This section describes some general **Shape Tools** features.



ACCESSING SHAPE TOOLS FUNCTIONS

The primary **Shape Tools** functions are accessible through buttons and serve the following main purposes:

Defining new shapes in a project: Create or import shapes in a project by selecting the shape function appropriate to the type of shape you want to create. For more information, see [“Creating Shapes” on page 8-10](#). Import existing shapes to a project from the Corda 7 Gallery. For more information, see [“The Corda 7 Gallery” on page 3-8](#).

Manipulating existing shapes: Once one or more shapes are in a project, Shape Tools lets you manipulate those shapes in several ways. Options include moving a shape; resizing a shape; modifying, distorting, and twisting a shape; and rotating one or more shapes. For more information, see [“Manipulating Shapes” on page 8-13](#).

- OTHER CORDA OBJECTS
- Shapes
-
-

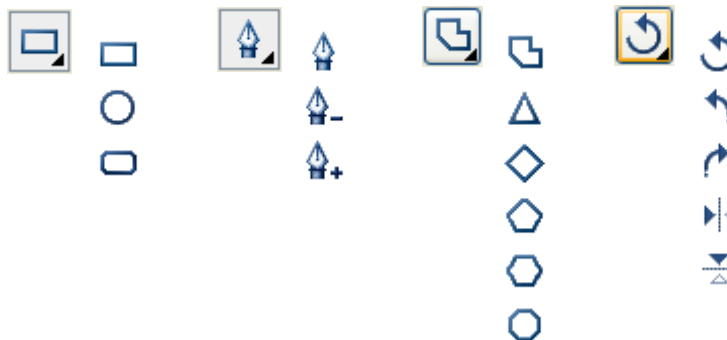
Creating text boxes: **Shape Tools** lets you draw text boxes on the Project Canvas, as opposed to inserting a pre-defined text box via **Create > Text box**. For more information about text boxes, see **“Text Boxes”** on page 8-5.

Creating radial gauge dials: **Shape Tools** also provides the ability to create radial gauge dials. For more information about gauges, see **“Gauges”** on page 4-25.

Important: *Shape Tools includes a small padlock icon in the header that lets you specify cursor behavior when using Shape Tools. When “locked,” Shape Tools reverts to the default **Select an object** function after each shape operation. When “unlocked,” the cursor remains as the currently selected function.*

SECONDARY MENUS

Some of the primary **Shape Tools** functions include secondary menus that provide additional features related to those functions. The presence of secondary menus are indicated by a black mark in the lower right corner. To activate the secondary menu, click and hold the primary function button.



SHAPE TOOLS MOUSE POINTERS

When working with shapes, **Corda Builder** uses different mouse pointers to indicate the type of action that is currently available:

Note: *The cursor images below are default Windows-based cursors, but cursors can vary based upon the operating system and its configuration.*

Move cursor: Clicking and dragging when this cursor is visible lets you move the currently selected shape or shapes.

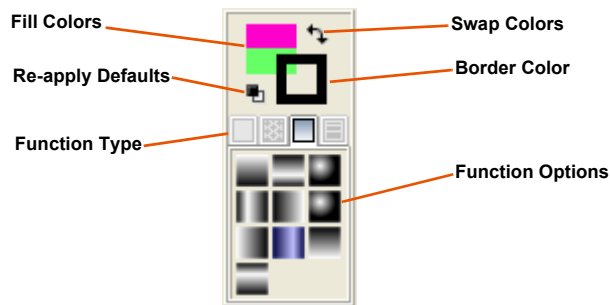
Resize cursor: Clicking and dragging when this cursor is visible lets you resize the currently selected shape.



+ **Single point cursor:** Clicking and dragging when this cursor is visible lets you manipulate a single point on the shape.

SHAPE FILL AND COLOR

Corda Builder **Shape Tools** provide robust and easily accessible functions for manipulating shape fill and color options. Corda Builder applies fill and color functions to all currently selected shapes.



The **Shape Tools** fill and color functions include the following:

Fill Color: Specifies primary and secondary fill colors for the selected shapes. Solid fill uses a single (primary) fill color, while pattern and gradient fill use a primary and a secondary fill color. To select a color, click the **Fill color** field, which opens the **Color Properties** dialog.

Border Color: Specifies a border color for the selected shapes. To select a color, click the **Border color** field, which opens the **Color Properties** dialog.

Swap Colors: Swaps the currently selected fill and border colors. When using two fill colors, the primary fill color swaps with the border color.

Re-apply Defaults: Replaces the current shape color scheme with the default color scheme, which is a solid beige fill with a black border.

Function Type: Specifies the fill type or border width of the selected objects. These four buttons let you specify the specific color characteristic with which you are working, which include Solid (single color) fill, Pattern fill, Gradient fill, and Border width. The selected function type determines the **Function Options** displayed below the **Function Type** buttons.






Function Options: Displays the options available for the currently selected **Function Type**. The Solid fill function displays the 12 most-recently used fill

- OTHER CORDA OBJECTS
- *Shapes*
-
-

colors. The Pattern fill function displays the available fill patterns. The Gradient fill function displays the available gradient fill types.

CREATING SHAPES

Corda Builder **Shape Tools** creates both generic and custom shapes:

Shape	Example	Category
Line		Generic
Rectangle (includes Circle and Rounded Rectangle)		Generic
Polyline		Custom
Polygon (includes Triangle, Diamond, Pentagon, Hexagon, and Octagon)		Custom
Text box		Generic

To create a shape other than those in the table above, Corda Builder **Shape Tools** let you manipulate one of these shapes or combine multiple shapes into a new shape. For more information, see [“Manipulating Shapes” on page 8-13](#), and [“Creating an Area Shape” on page 8-16](#).

Information on specific shape properties is available in the [Corda 7 Object Reference](#).

GENERIC SHAPES



Generic shapes include Lines, Ellipses, Rectangles, Rounded Rectangles, and Text boxes. For more information about Text boxes, see [“Text Boxes” on page 8-5](#).

To create a generic shape

- 1 **Select the desired shape type, either [Line](#), [Rectangle](#), or [Text box](#).**

Circle (Ellipse) and Rounded Rectangle shapes are in the Rectangle button's secondary menu.

- 2 **On the [Project Canvas](#), click and hold the left mouse button, and drag the mouse pointer to create the selected shape.**
- 3 **Release the mouse button to finalize the object on the [Project Canvas](#).**

Alternately, simply click the [Project Canvas](#) to create a generic shape of a standard size, as determined by Corda 7.

Note: *To force Corda Builder to maintain proportion for circle or square shapes, hold down the <Shift> key while dragging the mouse to create the shape.*

Once created, a generic shape can be moved and manipulated as needed. For more information about manipulating shapes, see ["Manipulating Shapes" on page 8-13](#).

CUSTOM SHAPES



Custom shapes are those that require additional user input during the creation process, and include Polygons and Polylines.

To create a Polygon or Polyline

- 1 **Select the desired shape type, either [Polygon](#) or [Polyline](#).**

When creating a polygon, double-click the [Polygon](#) button to open the [Polygon Tool](#), from which you can instruct Corda Builder to create a polygon of a specified number of sides.

- 2 **On the [Project Canvas](#), click the left mouse button to mark the starting point of the custom shape.**
- 3 **Move the mouse pointer to the location of the second vertex and click the left mouse button again. Repeat this process for each vertex in the custom shape.**
- 4 **Double-click the mouse on the final vertex to finalize the object on the [Project Canvas](#). Alternately, Corda Builder automatically finalizes the custom object if you click outside of the [Project Canvas](#).**

Note: *The Polygon shape button includes a secondary menu for creating some common polygon shapes, including [Triangle](#), [Diamond](#), [Pentagon](#), [Hexagon](#), and [Octagon](#). Selecting a shape from this secondary menu creates the selected shape in a fixed size that can then be re-sized and manipulated as needed.*

Once created, move and manipulate polygons and polylines as needed. For more information about manipulating shapes, see ["Manipulating Shapes" on page 8-13](#).

OTHER CORDA OBJECTS

Shapes

DIAL GAUGE SHAPE



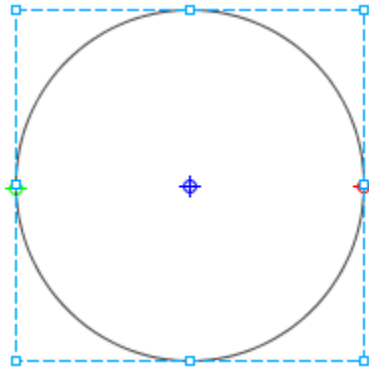
Corda Builder **Shape Tools** lets you configure a radial gauge shape as a framework for a radial gauge object. The resulting radial gauge shape defines key characteristics for the resulting radial gauge.

Note: *Corda 7 offers extremely powerful gauge creation options. For more information about creating and using gauges, see [“Gauges” on page 4-25](#).*

To create a radial gauge shape

- 1 Click the **Create a dial gauge shape** button.
- 2 On the **Project Canvas**, click and drag to create the initial dial shape.

The dial shape is a modified circle shape that includes three gauge-related handles. Once created, resize or reshape the dial shape as you would any other circle or ellipse. For more information, see [“Select an Object” on page 8-13](#).



- 3 Manipulate the three gauge handles to configure the dial shape as needed.

Blue handle (center): Marks the location of the dial shape’s spindle pivot point. This is where the dial spindle is anchored to the dial face.

Green handle (left): Marks the starting point for the dial shape’s scale.

Red handle (right): Marks the ending point for the dial shape’s scale.

Once the dial shape is created, there are additional steps for finishing the radial gauge. For information about creating a complete radial gauge, see [“Radial Gauges” on page 4-30](#).

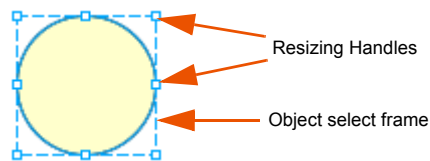
MANIPULATING SHAPES

Corda Builder **Shape Tools** includes the following functions for manipulating shapes on the Project Canvas. For introductory information about navigating the **Shape Tools** interface and using **Shape Tools** functions, see “Using Shape Tools” on page 8-7.

SELECT AN OBJECT



Select an object is the default **Shape Tools** function. This cursor lets you select an entire shape when you click it in the **Project Canvas**. This is equivalent to selecting a shape in the **Object list**. Selecting an object in this way surrounds it with a rectangular frame that includes resizing handles (small squares outlined in blue) on the corners and sides.



Two actions are available with the **Select an object** function:

- Click and drag the entire shape to re-position it within the **Project Canvas**. This is the action performed if the mouse pointer is placed anywhere on the shape other than the resizing handles.
- Click and drag one of the resizing handles on the shape’s frame to resize the object. The resizing handles on the sides of the shape’s frame only allow resizing along one axis, either horizontal or vertical. The resizing handles on the corners allow resizing along both axes at once.

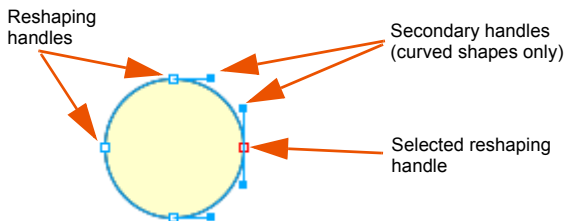
SELECT A POINT



Select a point allows you to select a shape for the purpose of modifying or distorting the shape. Selecting a shape in this way outlines it in a frame that

- OTHER CORDA OBJECTS
- Shapes
-
-

includes reshaping handles (small blue-outlined squares) along the shape’s border. A reshaping handle’s outline turns red to indicate it has been selected.



Two actions are available with the **Select a point** function:

- Click and drag the entire shape to re-position it within the Project Canvas. This is the action performed if the mouse pointer is placed anywhere on the shape other than the reshaping handles.
- Click and drag one of the reshaping handles on the shape’s frame to move that point only, thereby distorting the original shape in some way.

Note: *When working with grouped multi-shape objects, including maps, the **Select a point** function lets you select a single shape within the object to work with. This lets you manipulate individual shapes within the object without having to break the object apart.*

Important: *Curved shapes, such as Circles and Ellipses, provide secondary reshaping handles when you select a primary handle. These secondary handles are represented as solid blue squares connected to the primary reshaping handle by a blue line. Use the secondary handles for twisting and bending the shape.*

ADD AND REMOVE RESHAPING HANDLES

The **PolyLine/Polygon** button includes a secondary menu with two options for manipulating or configuring the handles associated with a shape.



Remove a point: The Remove a point function allows you to select a vertex (point) on a shape for the purpose of modifying or distorting the shape. Selecting a shape in this way outlines it in a frame that includes handles at each of the shape’s points.

To remove a point, simply select it. Corda Builder deletes that point and connects the points on either side of the deleted point with a straight line.

Note: *You cannot delete the anchor point (the first point defined) of **Polygons** or **Polylines** unless all other points have been removed first.*



Add a point: The Add a point function allows you to define a new handle at any location along the border of a straight-line shape. Once added, the new handle can be used with both the Select an object and the Select a point functions to resize or modify/distort the shape.

To add a point, simply click the shape border at the location where you want the new handle defined. Corda Builder adds a handle at that point.

Important: *The **Add a point** function is not currently available for curved shapes, such as circles, ellipses, and rounded rectangles.*

ROTATE SHAPES



The **Rotate selected shapes** function allows you to manually rotate one or more shapes around a rotation point that you define. Use the **Select an object** function to select the shapes you want to rotate before attempting to rotate them.

Two shapes, Rounded Rectangles and Lines, cannot be rotated by default. To rotate these shapes, Corda Builder lets you convert these shapes to Polygons. However, once converted, the new polygons do not have any of the configuration options specific to Rounded Rectangles (corner radius) or Lines (line ends such as arrows) shapes; these characteristics are fixed and cannot be changed.

To rotate selected shapes, complete the following steps:

- 1 **Left-click at any point in the Project Canvas to define the rotation point. Once selected, this point is defined with a cross-hair symbol** .
- 2 **Click and drag one of the selected shapes to cause it to rotate around the defined rotation point.**

The **Rotate selected shapes** button includes a secondary menu of some standard rotation operations, including **Rotate 90 degrees left**, **rotate 90 degrees right**, **Horizontal flip**, and **Vertical flip**. When you select a rotation option from this secondary menu, shapes are rotated immediately. No further action is needed.

Note: *Double-click the **Rotate selected shape** button to open the **Rotate Tool**, from which you can specify a specific number of degrees to rotate the shape. A positive number rotates the shape to the right. A negative number rotates the shape to the left.*

COPY SHAPE COLORS



The **Copy shape colors to another shape** function lets you synchronize a shape's color characteristics to one or more shapes. To copy a shape's fill to other shapes, complete the following steps:

- 1 **Select the shapes to which you want to copy color characteristics.**
- 2 **Click the **Copy shape colors to another shape** button.**
- 3 **Select the model shape from which you want to copy color characteristics.**

For more information about setting shape colors, see ["Shape Fill and Color" on page 8-9](#).

- OTHER CORDA OBJECTS
- Shapes
-
-

CREATING AN AREA SHAPE

Sometimes the simple, predefined shapes are not sufficient, so Corda Builder provides various tools and methods that allow you to create more complex shapes as part of the project. There are multiple ways to create complex shapes:

- [Grouping Shapes](#)
- [Combining Shapes](#)
- [Dividing a Shape](#)

GROUPING SHAPES

One way to create more complex shapes for the project is to group shapes together. Grouped shapes are treated as a unit for the purposes of positioning, sizing, and deleting. This makes it much easier to work with multiple related shapes as you create the Image Template file in Corda Builder.

However, grouped shapes can still be selected individually from the [Object List](#) in order to configure individual shape properties.

To group shapes

- 1 **Add the shapes you want to group to the project window.**
- 2 **Position and size the individual shapes into the desired configuration.**
- 3 **Select all shapes to be grouped together.**

To do this, hold down the <Ctrl> key while selecting each of the desired objects, or drag a frame around all the shapes on the [Project Canvas](#) that you want to group.

- 4 **Right-click the group of shapes and select [Combine into New Group](#).**

COMBINING SHAPES

Combining shapes differs from grouping objects; instead of maintaining a separate object identity for each shape involved in the complex shape, a single, indivisible shape is created. From the perspective of shape formatting, a combined shape is treated as a single object.

Remember that when multiple objects, including shapes, are added to a project, they are layered, meaning that a more recently added shape overlaps a previously added shape. However, Corda Builder lets you change the layering of objects as needed. For more information, see [“Aligning Objects” on page 3-4](#).

Note: *Because combining shapes creates a new shape object, a combined shape can be subsequently grouped with other shapes, if desired. For more information on grouping shapes, see [“Grouping Shapes” on page 8-16](#).*

There are multiple ways to combine shapes, including the following:

- [Unite](#)

- [Intersect](#)
- [MinusFront](#)
- [MinusBack](#)

UNITE

Unite is the most straightforward of the tools for uniting shapes. It functions in the same way as grouping objects. But instead of maintaining each component shape as a separate object in the [Object List](#), a single new shape object is formed that replaces all the component shapes in the [Object List](#).

To combine shapes with Unite

- 1 **Add the shapes you want to group to the [Project Canvas](#).**
- 2 **Position and size the individual shapes into the desired configuration.**
- 3 **Select all shapes to be combined.**

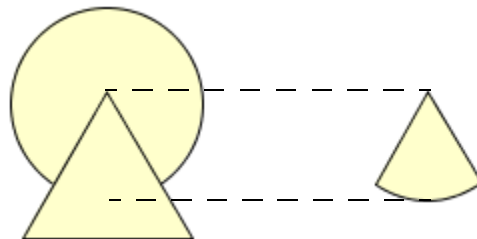
To do this, hold down the <Ctrl> key while selecting each of the desired objects, or drag a frame around all the shapes you want to group on the [Project Canvas](#).

- 4 **Select [Properties > Unite](#).**
- 5 **Specify the name of the new shape and click [OK](#).**

If you want the inner border lines between the component shapes removed in the new shape, check [Remove inner lines](#).

INTERSECT

Intersect creates a new shape in the form of the intersection, or overlapping portion, of two component shapes.



OTHER CORDA OBJECTS*Shapes***To combine shapes with Intersect**

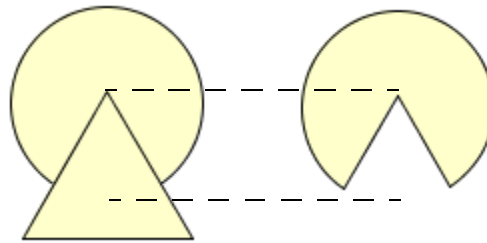
- 1 Add the shapes you want to group to the **Project Canvas**.
- 2 Position and size the individual shapes into the desired configuration.
- 3 Select the two shapes to be combined.

To do this, hold down the <Ctrl> key while selecting each of the desired objects, or drag a frame around all the shapes on the **Project Canvas** that you want to group.

- 4 Select **Properties > Intersect**.
- 5 Specify the name of the new shape and click **OK**.

MINUSFRONT

MinusFront creates new objects by “stamping out” the intersection between two or more component shapes. The backmost shape layer is used as the pattern shape. Any overlapping shapes in front of the pattern shape are removed from the pattern shape.

**To combine shapes with MinusFront**

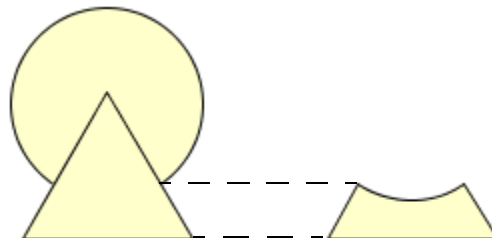
- 1 Add the shapes you want to group to the **Project Canvas**.
- 2 Position and size the individual shapes into the desired configuration.
- 3 Select the shapes to be combined.

To do this, hold down the <Ctrl> key while selecting each of the desired objects, or drag a frame around all the shapes on the **Project Canvas** that you want to group.

- 4 Select **Properties > MinusFront**.
- 5 Specify the name of the new shape and click **OK**.

MINUSBACK

MinusBack creates new objects by “stamping out” the intersection between two or more component shapes. The front-most shape layer is used as the pattern shape. Any overlapped shapes behind the pattern shape are removed.



To combine shapes with MinusBack

- 1 Add the shapes you want to group to the **Project Canvas**.
- 2 Position and size the individual shapes into the desired configuration.
- 3 Select the shapes to be combined.

To do this, hold down the <Ctrl> key while selecting each of the desired objects, or drag a frame around all the shapes on the **Project Canvas** that you want to group.

- 4 Select **Properties > MinusBack**.
- 5 Specify the name of the new shape and click **OK**.

DIVIDING A SHAPE

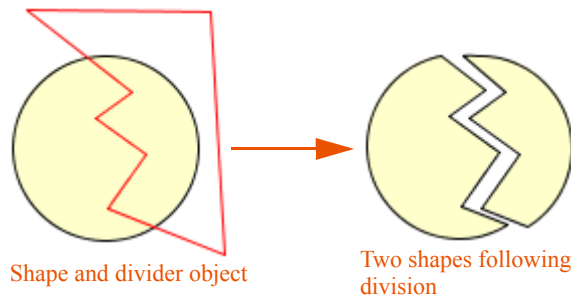
In addition to combining shape objects into new shapes, you can also divide a single shape along a border defined by another shape.



- OTHER CORDA OBJECTS
- *Shapes*
-
-

To divide a shape

- 1 Add the shape you want to divide to the **Project Canvas**.
- 2 Add a shape to act as a divider object to the **Project Canvas**.
- 3 Select the object and select **Properties > Convert to Divider Object**.
- 4 Position the divider object over the shape to be divided.
- 5 Select both the divider shape and the shape to be divided.



- 6 Select **Properties > Divide shapes**.

Following the shape division, there are two separate objects in the **Object List**, representing the two portions of the divided object. These are now fully autonomous shapes that can be positioned, sized, and formatted independently.

IMAGES

Imported images allow you to accomplish such tasks as including a corporate logo in an Image Template file, or using an image as a background to a graph or map. Corda Builder encodes imported images into the project. Imported graphics can be any image file of the following formats: GIF, PNG, and JPEG.

Imported images work a lot like any other object in Corda Builder. Add a border, add drilldown effects, add descriptive text, and change the image in **Object Properties**.

Warning: Adding an imported image can significantly increase the size of an Image Template file. If you intend to use it with Corda Server™, a very large imported image can significantly affect image display time.



Import, move, and delete images as you would any other object in a Corda Builder project. For more information, see [“Working with Objects” on page 3-2](#).

This section includes the following:

- [Image Resizing](#)
- [Image Transparency](#)
- [Image Borders and Shadows](#)

- OTHER CORDA OBJECTS

- *Images*

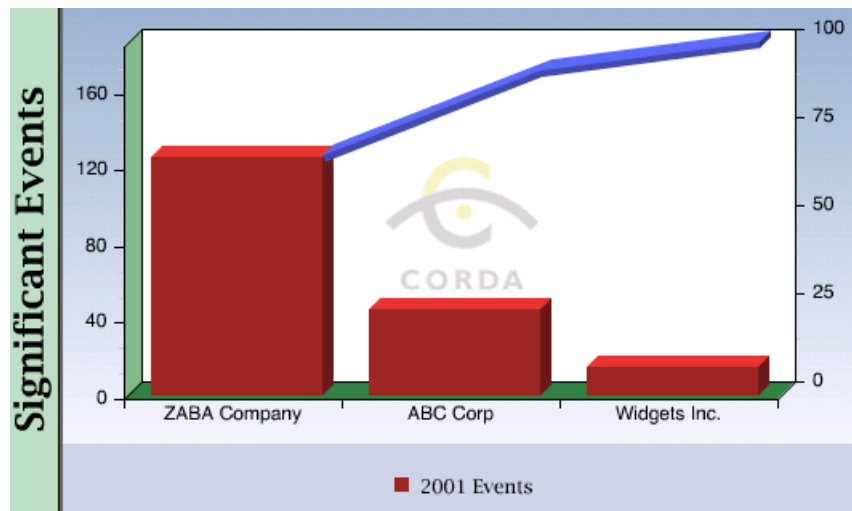
IMAGE RESIZING

Like any other object, resize imported graphics by selecting a corner of the image and dragging it in or out. However, since an imported graphic is scaled as it is resized, the image's quality can be adversely affected.

Properties and attributes related to image resizing are available from the [Common](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

IMAGE TRANSPARENCY

To use an imported image as a watermark or to place an imported image on top of graphs or maps and still have objects visible behind it, set the transparency of the imported object.



Properties and attributes related to image transparency are available from the [Image Settings](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

IMAGE BORDERS AND SHADOWS

To further accentuate the imported image, add border and shadow effects.

Properties and attributes related to image border and shadow are available from the *Image Settings* property in *Object Properties*. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- **OTHER CORDA OBJECTS**
- *Images*
-
-



DRILLDOWN EFFECTS

Using the Corda® Builder™ drilldown capabilities lets you associate URL links or Javascript functions with data items in the Corda image. This functionality is often used to allow a user to navigate from one Corda image to another. The user can select a data item (such as a bar or pie wedge) in a graph or map on one Web page and be brought to a different Web page that provides expanded information on the data item.

This chapter introduces drilldown concepts and how to add drilldown to Corda objects. It includes the following sections:

- [Viewing Drilldown](#)
- [Identifying Drilldown](#)
- [Graph Drilldown](#)
- [Map Drilldown](#)
- [Other Object Drilldown](#)
- [Drilldown Meta Tags](#)
- [Javascript Drilldown](#)

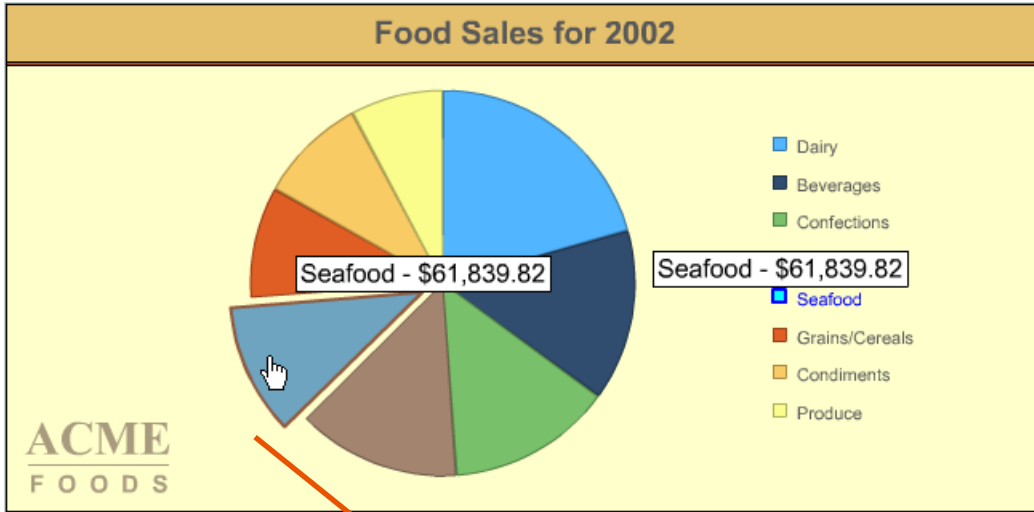
Note: *Drilldown is supported in FLASH, SVG, JPEG, and PNG images only. Drilldown may be unavailable in JPEG or PNG images with some browsers. For more details, see [“Drilldown / Hover / Roll-Over Data Label Problems with PNG or JPEG Images”](#) on page 11-11 of the Corda Installation and Administration Guide.*

Important: *Drilldown isn't displayed in Corda Builder. To view a drilldown, use Corda Builder's Preview option, or view the saved Corda image through a Web browser. For more information, see [“Viewing Drilldown”](#) on page 9-3.*

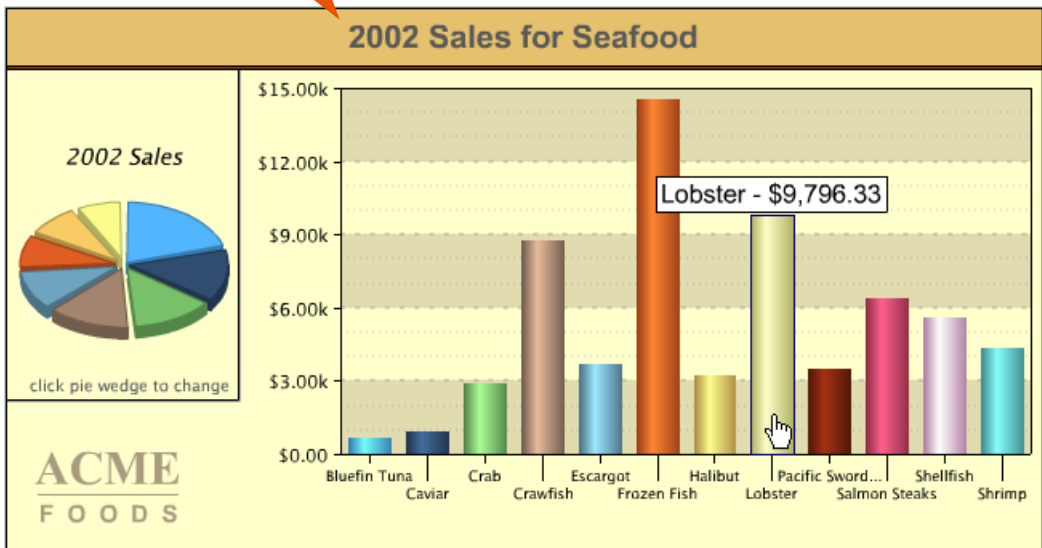
For example, suppose a Corda image contains a graph that represents the annual sales for each of a company's products. Each data item in that graph contains a drilldown link to

DRILLDOWN EFFECTS

another Corda image that breaks down the sales for the corresponding product. Users can access the link to view the more detailed information, and then return to the original graph.




Selecting a pie wedge opens another graph with more specific information.



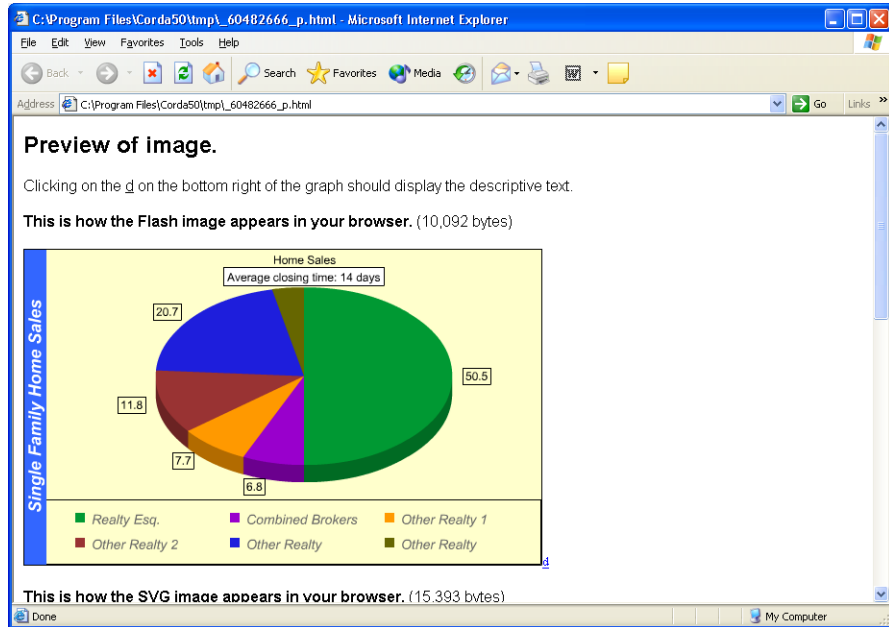
VIEWING DRILLDOWN

Because drilldown does not show in the Corda Builder interface, preview the Corda image to see drilldown effects.

To preview drilldown

- 1 Select **File > Preview**, or click the **Preview**  button on the toolbar.

The system's default Web browser opens and displays the Corda image.



- 2 View the drilldown by mousing over the Corda Image.

- DRILLDOWN EFFECTS
- *Identifying Drilldown*
-
-
-

IDENTIFYING DRILLDOWN

Corda images in Flash or SVG formats highlight drilldown-enabled data items as a user mouses over them. Depending on the image type, the Web browser's status bar might display the associated URL. If the user selects the highlighted data item, the Web browser performs the associated drilldown action.

Note: *When a mouse pointer passes over a drilldown-enabled data item in a JPEG or PNG image, the mouse pointer turns into a hand, but the data item is not highlighted.*

Properties and attributes related to identifying drilldown are available from the `Drilldown` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

GRAPH DRILLDOWN

Static drilldown effects can be added to the following graph components:

- [Default Graph Drilldown](#)
- [Data Item Drilldown](#)

For information about adding drilldown effects dynamically, see Chapter 6, “[Building Drill-down](#),” of the *Corda 7 Developer Reference*.

DEFAULT GRAPH DRILLDOWN

Default graph drilldowns let you configure a common drilldown destination for all the data items in the graph. Global drilldown destinations are applied to any data item for which a drilldown override isn’t specified.

Specify the default graph drilldown in the `URL` attribute of the `drilldown` property. More information on these drilldown properties and attributes is available in the [Corda 7 Object Reference](#).

For example, if you are drilling down to a Web application that accepts data values on the query string, a data item’s drilldown URL might look something like the following:

```
http://webapp.mycompany.com/database.cgi?series=Red&category=November
```

Instead of hard-coding the URL and a unique query string value for each data item in the graph, specify a default graph drilldown using meta tags to dynamically insert the custom query string for each data item. For example:

```
http://webapp.mycompany.com/database.cgi?series=%_SERIES_NAME&category=%_CATEGORY_NAME.
```

For more information on drilldown meta tags, see “[Drilldown Meta Tags](#)” on page 9-10.

Note: *This is only an example of how a default drilldown might be configured. Corda Builder meta tags are flexible enough to fit almost any potential situation.*

DATA ITEM DRILLDOWN

If desired, Corda Builder lets you add drilldown effects to individual graph data items.

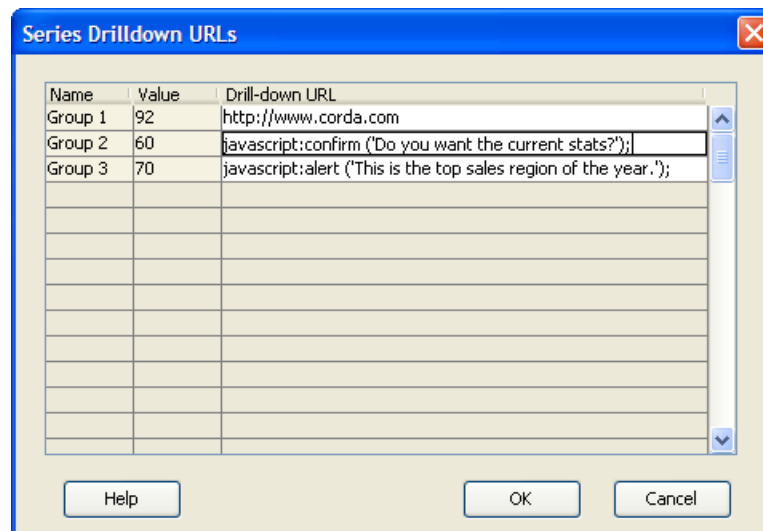
- DRILLDOWN EFFECTS
- *Graph Drilldown*
-
-

To add a data item drilldown

- 1 Select the graph.
- 2 Locate the appropriate data series in **Object Properties**.
- 3 In the **URL** property, click the ellipsis [...] button to open the **Series Drilldown URLs** dialog.
- 4 Enter the desired drilldown URL or Javascript in the **Drill-down URL** field for the appropriate data item(s).

To help identify each data item, the **Name** and **Value** columns indicate the category to which the data item belongs and the value of the data item.

Place user-defined Javascript functions on the Web page to which you are adding the Image Template file, and then call those Javascript functions from the drilldown link by using Javascript. For example: `Javascript:function-name(parameter-list)` calls a function on the Web page named `function-name`.



More information on drilldown properties and attributes is available in the [Corda 7 Object Reference](#).

Note: *Corda Builder* automatically inserts the required `http://` at the beginning of a URL. If the link is to a secure URL, make sure to specify `https://` as part of the URL.

MAP DRILLDOWN

Map drilldowns can be added in the following ways:

- [Map Layer Drilldown](#)
- [Map Range and Data Item Drilldown](#)

For information about adding drilldown effects dynamically, see Chapter 6, “[Building Drill-down](#),” of the *Corda 7 Developer Reference*.

Note: For more information on map layers, see “[Working with Layers](#)” on page 6-16.

MAP LAYER DRILLDOWN

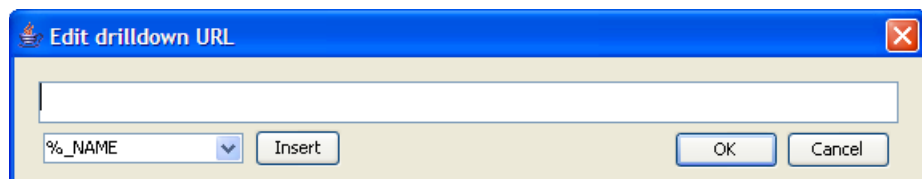
The easiest way to add drilldown effects to a map is to do so globally. Accessing a map layer’s default shape properties lets you quickly add drilldown to all of the data items in a map layer at once. Through the use of meta tags, links can be different for each data item, changing based on the data item’s name, range, and value. For more information on meta tags, see “[Drilldown Meta Tags](#)” on page 9-10.

Map layer drilldowns let you configure a common drilldown destination for all the data items in the map. Default drilldown destinations are applied to any data item for which a drilldown override isn’t specified.

To add a map layer drilldown

- 1 **Select the map layer in the Object List.**
- 2 **In [Default Shape > Drilldown](#), select the [URL](#) property and click the ellipsis [...] button to open the [Edit Drilldown URL](#) dialog.**
- 3 **Enter the desired drilldown URL or Javascript in the URL field.**

Specify any combination of text and meta tags to define the drilldown effect. Select supported meta tags to include in the drilldown from the dropdown list, if desired. For more information on drilldown meta tags, see “[Drilldown Meta Tags](#)” on page 9-10.



More information on drilldown properties and attributes is available in the [Corda 7 Object Reference](#).

- DRILLDOWN EFFECTS
- *Map Drilldown*
-
-

MAP RANGE AND DATA ITEM DRILLDOWN

Corda Builder lets you add map drilldowns to map ranges or to individual map data items, although these are less powerful options than adding a global effect. These drilldowns override any default map layer drilldowns that might be specified.

To add a map range or data item drilldown

- 1 **Select the appropriate map range or map shape.**

Select map ranges from [Object Properties](#). Select individual map shapes from the [Object List](#).

- 2 **Select the [url](#) property and click the ellipsis [...] button to open the [Edit Drilldown URL](#) dialog.**

For map ranges, the drilldown is defined in the [Range Drilldown](#) property, where *Range* is the [Range](#) property to which you want to add a drilldown effect. For individual map shapes, the drilldown is defined in the [map-shape > drilldown > url](#) property.

- 3 **Enter the desired drilldown URL or Javascript in the URL field.**

Specify any combination of text and meta tags to define the drilldown effect. Select supported meta tags to include in the drilldown from the dropdown list, if desired. For more information on drilldown meta tags, see [“Drilldown Meta Tags” on page 9-10](#).

More information on drilldown properties and attributes is available in the [Corda 7 Object Reference](#).

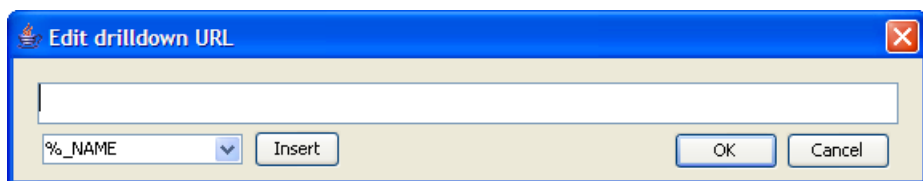
OTHER OBJECT DRILLDOWN

Shape, Imported Image, and Textbox also support drilldown through the **drilldown** property.

To add drilldown to a shape, image, or textbox

- 1 Select the appropriate object in the **Object List**.
- 2 In the **Drilldown > URL** property, click the ellipsis [...] button to open the **Edit Drilldown URL** dialog.
- 3 Enter the desired drilldown URL or Javascript in the URL field.

Specify any combination of text and meta tags to define the drilldown effect. Select supported meta tags to include in the drilldown from the dropdown list, if desired. For more information on drilldown meta tags, see [“Drilldown Meta Tags” on page 9-10](#).



More information on drilldown properties and attributes is available in the [Corda 7 Object Reference](#).

- DRILLDOWN EFFECTS
- *Drilldown Meta Tags*
-
-

DRILLDOWN META TAGS

Like data labels, drilldown definitions support meta tags. In many cases, this makes it possible to define a single default drilldown and dynamically apply data as the user mouses over different data items in the Corda Image.

For example, setting a graph's drilldown definition to `morestats.jsp?%_SERIES_NAME` causes the graph to drilldown to `morestats.jsp?Blue` when a user selects any data item in the Blue data series.

Similarly, setting a map's drilldown definition to `morestats.jsp?%_NAME` causes the map to drilldown to `morestats.jsp?Japan` when a user selects a map shape named *Japan*.

Many objects include a meta tag dropdown list in the Edit Drilldown dialog from which to select valid meta tags to include in the drilldown definition.

For a complete list of supported meta tags, see [“Graph Meta Tags” on page 10-5](#).

Note: *For information about dynamically adding or changing drilldown definitions, see Chapter 6, “Building Drill-down,” of the Corda 7 Developer Reference.*

JAVASCRIPT DRILLDOWN

In addition to URL drilldowns, Corda Builder supports Javascript drilldown, which runs a custom Javascript function when a user selects drilldown data item. The following topics describe how to use Corda Builder's Javascript drilldown capabilities:

- [Drilling Down to Basic Functions](#)
- [Opening a New Window](#)
- [Encoding Issues with Extended Characters](#)

DRILLING DOWN TO BASIC FUNCTIONS

To drill down to a Javascript function, specify the appropriate Javascript in the drilldown definition: `Javascript:function()`, where `function()` is the name of the function to run. This function must be a built-in Javascript function or a function defined in the Web page.

For example, the `alert()` function is a built-in Javascript function that pops up a system message box. Drill down to the alert function by entering code similar to the following in the drilldown string:

```
Javascript:alert('%_NAME=%_VALUE')
```

This same technique works with any Javascript function that you define in the Web page.

OPENING A NEW WINDOW

A common task that can be performed with Javascript drilldown is to open a link in a new window. This can be done using the predefined `window.open()` Javascript function. The example below shows how to use this function to set a global drilldown string.

```
Javascript:void window.open  
('http://www.myserver.com/getgraph?id=%_VALUE')
```

Important: *Note the use of `void` at the beginning of the statement. This is necessary to prevent the current browser window from browsing to an empty object.*

Note: *`Object Properties` also lets you open the drilldown in a new window by checking the `Open in New Window` attribute of the `Drilldown` property.*

- DRILLDOWN EFFECTS
- *Javascript Drilldown*
-
-
-

The `window.open()` command also allows you to control the size and attributes of the new window. The example below shows how to take advantage of this functionality.

```
void window.open(
  'http://www.myserver.com/getgraph/?id=%_VALUE',
  'WindowName',
  'directories=no,height=680,width=680,hotkeys=no,menu
  bar=no,location=no,personalbar=no,status=no,toolbar=
  no,scrollbars=yes');
```

Refer to a Javascript manual for more information about `window.open()`.

ENCODING ISSUES WITH EXTENDED CHARACTERS

By default, Corda Builder URL-encodes all drilldown effects. In other words, drilldown strings are encoded in such a manner that browsers are able to properly interpret the intended drilldown destination.

Unfortunately, this creates a problem when the intended target of the drilldown effect is a Javascript function instead of a URL. Corda Builder may URL-encode characters that should be Javascript-encoded instead. Because Javascript functions do not recognize URL encoding, characters might display improperly.

This situation is further complicated on occasions when strings in Javascript functions should be URL-encoded because the string is eventually interpreted by a browser as a URL. Because this is often the case, Corda Builder cannot simply attempt to detect whether a drilldown definition is a URL or a Javascript function. By default, anytime a drilldown passes a string containing an extended character (higher than 127 (0x7E)) through a Javascript function, the string is interpreted as a URL.

For those situations when Corda Builder should interpret this type of drilldown string as Javascript, use the `%_JS_ENCODE` meta tag. This meta tag instructs Corda Builder to Javascript-encode the drilldown effect rather than URL-encode it. Include the `%_JS_ENCODE` tag at the end of any drilldown definition that needs to be Javascript-encoded. For example, the following drilldown definition doesn't display correctly. Instead of *Group™*, you see *Group %e2%84%a2*.

```
Javascript:alert("Category: Group ™")
```

To resolve this, add the `%_JS_ENCODE` meta tag, as follows:

```
Javascript:alert("Category: Group ™ %_JS_ENCODE")
```

DATA ANNOTATIONS

This chapter discusses ways to enhance the readability of a graph or map through data annotations. Adding data labels, hover text, and notes helps you to better describe the information that maps and graphs present.

This chapter includes the following sections:

- [Data Labels](#)
- [Hover Text](#)
- [Graph Notes](#)

DATA ANNOTATIONS

Data Labels

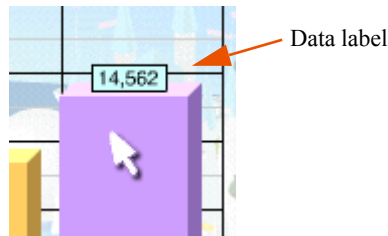
DATA LABELS

A data label is a small box of text associated with a data item (such as a **graph bar** or **shape**). Data labels can be displayed in one of two ways:

ALWAYS: Specifies that data labels are always visible.

ON ROLLOVER: Specifies that data labels are only visible when the mouse pointer passes over the associated data item. Rollover data labels are supported in FLASH, SVG, JPEG, and PNG images only. For more information, see [“Drilldown / Hover / Roll-Over Data Label Problems with PNG or JPEG Images”](#) on [page 11-11](#) of the *Corda Server Installation and Administration Guide*.

Note: Rollover data labels display the letters **RO** (rollover) in **Corda® Builder™** at the spot the data labels appear. These letters do not appear in the final Corda image. When you view the Corda image on the Web, the data labels only display when the mouse passes over an item (e.g., pie wedge or bar) in the graph.



Warning: Hover text and rollover labels cannot be used on the same item. Hover text always overrides the rollover data label. If you want the data label to appear even when there is hover text, configure the data label as a static data label.

Properties and attributes related to data labels are available from the **Data Label** property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

The remainder of this section discusses the following topics:

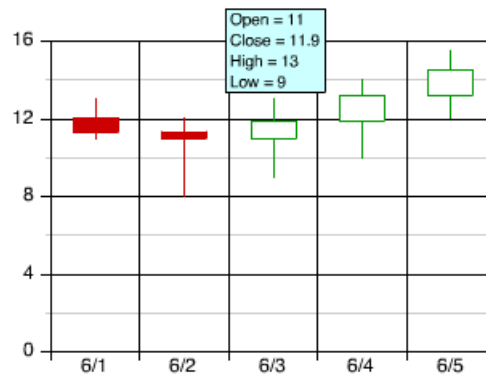
- [Data Label Text](#)
- [General Data Label Settings](#)
- [Graph Data Labels](#)
- [Map Data Labels](#)

DATA LABEL TEXT

The key to effective data labels is proper formatting of the data label text. By default, Corda Builder selects how data label text is formatted. Use meta tags to customize the format of data labels, providing complete control over what is displayed by a data label.

A meta tag is a keyword that begins with `%_`, such as `%_VALUE`. The meta tag is replaced by a dynamic value when the graph is published as an image. In this way, meta tags let you make sure each data label contains information specific to the data item with which it is associated.

Using a meta tag, a data label can display *Price: \$43* instead of just *43*. When combined with a data label's rollover capability, customized data label formatting can greatly reduce the need for hover text.



The data label format string can contain any combination of regular text and meta tags. The list of supported meta tags varies by object type, but all supported meta tags for a given object type are listed in the dropdown menu of the appropriate data label text property.

Note: Corda Builder supports the `
` tag to start a new line in data labels.

DATA VALUE FORMAT

Customize the format of the data displayed with value meta tags such as `%_VALUE` and `%_PERCENT_OF_CATEGORY`.

Note: Settings on this tab only apply to text in *Labels*, *Drilldown*, and *Hover Text* that use the `%_VALUE` keyword.

Properties and attributes related to data value format are available from the [Data Label](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- **DATA ANNOTATIONS**

- *Data Labels*
-
-

GENERAL DATA LABEL SETTINGS

Data labels support several common object properties, including the following:

- Border color and visibility
- Fill color
- Label margins
- Fixed label width
- Font and font characteristics
- Shadow visibility, color, and size

Properties and attributes related to data value format are available from the `Data Label` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

GRAPH DATA LABELS

Configure graph data labels in [Object Properties](#). The following configuration options are available for data labels:

- [Graph Data Label Position](#)
- [Graph Meta Tags](#)
- [Data Labels for Pareto Graphs](#)

GRAPH DATA LABEL POSITION

Corda Builder lets you specify the location of data labels within the graph, with specific positioning options dependent on graph type. It also lets you specify a fixed position at which all rollover data labels appear. A text box with the letters *RO* appears in the Image Template file. This is a placeholder for data labels. The data labels appear on rollover in the same condition and with the same format as this text box.

Properties and attributes related to data label position are available from the `Data Label` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

GRAPH META TAGS

Graph data labels: support the following meta tags:

meta tag	Description	Graphs
_%BUBBLE_VALUE	The value of the bubble in an X-Y or Time Bubble graph.	X-Y or Time Bubble
_%CATEGORY_NAME	The name of the category that the data item belongs to.	All, except X-Y and Time Plot
_%CATEGORY_NUMBER	The number of the data series that the data item belongs to.	All, except X-Y and Time Plot
_%CATEGORY_TOTAL	The sum of all data values in the category to which the data item belongs (applies to).	Area, Bar, Line, Pareto, Pie, Radar
_%CLOSE_VALUE	The close value for a high-low data item.	Stock
_%DESCRIPTION	The description of the data item.	All, except X-Y and Time Plot
_%FIRST_QUARTILE	The first quartile value for a box plot.	Box Plot
_%GRAPH_TOTAL	The sum of all data values in a bar, line, pie, or radar graph.	Area, Bar, Line, Pareto, Pie, Radar
_%HIGH_VALUE	The high value for a high-low data item.	Stock, Box Plot
_%LOW_VALUE	The low value for a high-low data item.	Stock, Box Plot
_%MEDIAN	The median value for a box plot.	Box Plot
_%CNAME	This meta tag is used in conjunction with a data file that provides the desired label text. For more information about using these data files, see "Comma-Separated Value Files" on page 2-10 of the <i>Corda 7 Developer Reference</i> .	All
_%OUTLIERS	A semi-colon separated list of all outlier values for a box plot.	Box Plot
_%OPEN_VALUE	The open value for a high-low data item.	Stock
_%PERCENT_OF_CATEGORY	The data value represented as a percentage of the sum of all data values in its category.	Area, Bar, Line, Pareto, Pie, Radar
_%PERCENT_OF_TOTAL	The data value represented as a percentage of the sum of all data values in the graph.	Area, Bar, Line, Pareto, Pie, Radar
_%POINT_NUMBER	The number of the data item in the data series (e.g., the first plot point=1).	X-Y and Time Plot

- DATA ANNOTATIONS

- Data Labels

meta tag	Description	Graphs
_%_SERIES_NAME	The name of the data series that the data item belongs to.	All but Stock, Box Plot
_%_SERIES_NUMBER	The number of the data series that the data item belongs to.	All but Stock, Box Plot
_%_THIRD_QUARTILE	The third quartile value for a box plot.	Box Plot
_%_TIME_VALUE	The time value for a Time Plot data item.	Time Plot
_%_VALUE	The value of the data item.	Area, Bar, Line, Pareto, Pie, Radar
_%_XVALUE	The x value for an X-Y data item.	X-Y
_%_YVALUE	The y value for an X-Y or Time Plot data item.	X-Y, Time Plot

DATA LABELS FOR PARETO GRAPHS

Data labels on a Pareto graph work just like data labels from any other graph. However, since the bars typically indicate a different type of value (e.g., dollars) than the line (percentage), you need to “trick” the graph into displaying a different symbol for the data labels along the line.

For example, suppose you set the data label format to `$_%_VALUE`. The bar series data labels display correctly (e.g., `$43,000`), but the line data labels, which should display as percentages, also include the dollar sign. Rather than set the general data label format, override the general data label format only for the pareto graph’s bar series, as shown in the following task:

To set a bar series data label

- 1 From the **Project Canvas** or the **Object List**, select the Pareto graph.
- 2 In **Object Properties**, open the **Bar Series** property.
- 3 Select the **Enable** box next to the **Data Label** attribute.
- 4 Set the **Data Label** value to the data label format you want to use for the bar series (for example, `$_%_VALUE%`).

DATA LABELS FOR PIE CHARTS

Pie charts are the only graphs that provides two data labels. Each is independently configurable. By default, the **Data Label** property is displayed, while the **Data Label 2** property is hidden.

DYNAMIC DATA LABELS

When using an Image Template file with Corda Server, data labels are often displayed dynamically in graphs, based on the live data. These dynamic data labels override those defined in Corda Builder. Enable data labels dynamically with ITXML.

To display a data label in a graph dynamically with ITXML, set the `enable-show` attribute of the `cit:data-label` property to `True`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

Note: *The `show` attribute of `cit:data-label` lets you configure the data label as either `always-on` or `on-hover`. The default setting is `on-hover`. Hover data labels are supported in FLASH, SVG, JPEG, and PNG images only. Hover may be unavailable in JPEG or PNG images with some Web browsers. For more details, see “[Drilldown / Hover / Roll-Over Data Label Problems with PNG or JPEG Images](#)” on page 11-11 of the *Corda 7 Install and Administration Guide*.*

Customize the look of a graph data label dynamically using either ITXML or PCScript:

ITXML

To customize a graph data label dynamically with ITXML, set the `text` attribute of the `cit:data-label` property. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

PCSCRIPT

To customize a graph data label dynamically with PCScript, use the `graph.SetDataLabelFormat()` method.

MAP DATA LABELS

Note: *Maps only support always-on data labels.*

Map data labels help identify and provide additional information about map shapes. Within maps, data labels can be applied at the following points:

- Map layer: `Label` property in `Default Shape`.
- Range: `Label` property in the appropriate `Range`.
- Map Shape: `Label` property in `Map Point` or `Map Shape`.

Note: *Since map shapes are just special instances of shape objects, data labels for regular shape objects can be configured in much the same way.*

- DATA ANNOTATIONS

- Data Labels

MAP META TAGS

Maps support the following meta tags:

_%CODE	The map data item code. This is one of three ways to identify a map data item.
_%DESCRIPTION	The value of %_DESCRIPTION is set from the ITXML tab. Select the Verbose XML option. You see a Description field associated with each map shape.
_%EXTRAINFO	The value of %_EXTRAINFO is set from the ITXML tab. Select the Verbose XML option. You see an ExtraInfo field associated with each map shape.
_%LONGNAME	The map data item's long name. This is one of three ways to identify a map data item.
_%CNAME	This meta tag is used in conjunction with a data file that provides the desired label text. For more information about using these data files, see "Comma-Separated Value Files" on page 2-10 of the <i>Corda 7 Developer Reference</i> .
_%NAME	The name of the data item. This is one of three ways to identify a map data item.
_%NAMECLASS	The data item type (e.g., city, state, province, zip code).
_%RANGENAME	The name of the range into which the data item falls.
_%PARENT	The name of the map to which the data item belongs.
_%VALUE	The value of the data item.
_%VALUENOFORMAT	The value of the data item, excluding any additional formatting.

MAP DATA LABEL LAYOUT

To configure a map's data label, use the `Layout` property in `Label`, which provides control over the label's position, margin, and maximum width.

Properties and attributes related to map data label positions are available from the `Layout` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

JUSTIFICATION

Since shape data labels automatically scale to the width of the label text, no justification is necessary for data labels with a single line of text. However, you may want to specify the text justification for multiple lines of text.

Properties and attributes related to map label justification are available from the `Label` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

DYNAMIC DATA LABELS

When using an Image Template file with Corda Server, data labels are often displayed dynamically in maps, based on the live data. These dynamic data labels override those defined in Corda Builder.

To display a dynamic data label in a map with ITXML, add a `cit:label` property to the appropriate `cit:default-shape`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

Use meta tags to change data label formats for map shapes in a map.

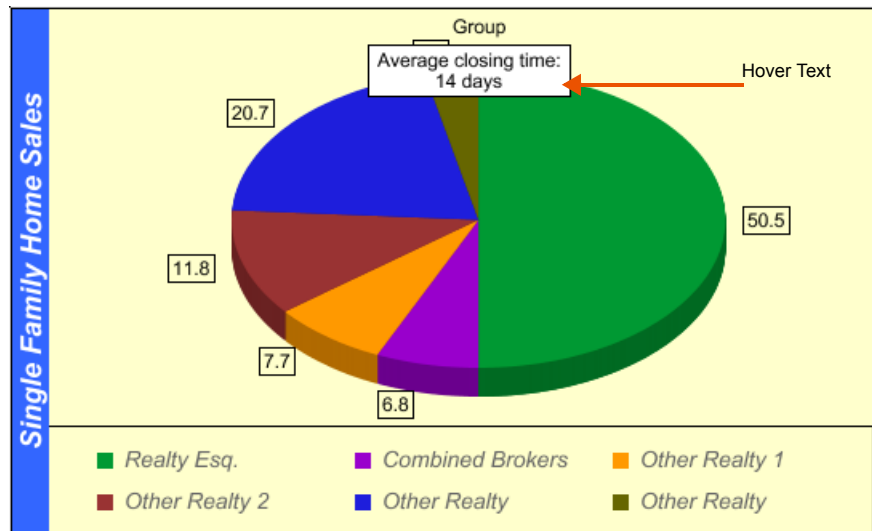
- DATA ANNOTATIONS
- Hover Text
-
-

HOVER TEXT

Hover text is used to display additional or supplemental information about a data item in a graph or map. Hover text is called such because it “pops” up as the user mouses over an item in a graph or map.

Important: *Hover text is supported in FLASH, SVG, JPEG, and PNG images only. Hover text may be unavailable in JPEG or PNG images with some browsers. For more details, see [“Drilldown / Hover / Roll-Over Data Label Problems with PNG or JPEG Images”](#) on page 11-11 of the Corda 7 Install and Administration Guide.*

For example, the following graph represents a realty company’s sales for homes in a given area and compares their sales to those of their competitor. The hover text provides information on the average closing time for each realty company in the chart and is displayed when the user mouses over each pie wedge.



Hover text is similar to data labels (see [“Data Labels”](#) on page 10-2) in many ways, particularly rollover data labels. The difference between data labels and hover text is that data labels automatically display information related to the data item’s value. Hover text, on the other hand, can display anything at all, but it must be manually specified for each data item. Because of this, hover text is typically used for data-specific annotations rather than general labels.

Warning: *Hover text and data labels cannot be used together unless the data label display option is set to Always. Enabling hover text for a data item prevents rollover data labels from displaying.*

Properties and attributes related to hover text are available from the `Hover Text` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

The remainder of this section discusses the following topics:

- [General Hover Text Settings](#)
- [Graph Hover Text](#)
- [Map Hover Text](#)
- [Shape Hover Text](#)
- [Viewing Hover Text](#)

Hover text isn't displayed in Corda Builder. To view hover text, use Corda Builder's Preview option, or view the saved Corda image through a Web browser.

GENERAL HOVER TEXT SETTINGS

Hover text is displayed in what is essentially a text box and can be configured as such. Corda Builder provides the following general hover text configuration parameters:

- Border color and visibility
- Fill color
- Hover textbox margins
- Fixed hover textbox width
- Font and font characteristics
- Shadow visibility, color, and size

Properties and attributes related to general hover text settings are available from the `Hover Text` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

HOVER TEXT FORMAT

The key to effective hover text is proper formatting. Meta tags can customize the format of hover text.

A meta tag is a keyword that begins with `%_`, such as `%_VALUE`. The meta tag is replaced by a dynamic value when the graph is published as an image. Using meta tags ensures that hover text contains information specific to the data item with which it is associated.

- DATA ANNOTATIONS

- *Hover Text*

The hover text format string can contain any combination of regular text and meta tags. The list of supported meta tags varies by object type, but all supported meta tags for a given object type are listed in the dropdown menu of the appropriate text property.

Note: *Corda Builder supports the `
` tag to start a new line in hover text.*

DATA VALUE FORMAT

Corda Builder lets you customize the format of the data displayed with value meta tags such as `%_VALUE` and `%_PERCENT_OF_CATEGORY`.

Properties and attributes related to general hover text settings are available from the `Hover Text` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

GRAPH HOVER TEXT

All data items (for example, bar, pie wedge, or X-Y point) in a graph support hover text.

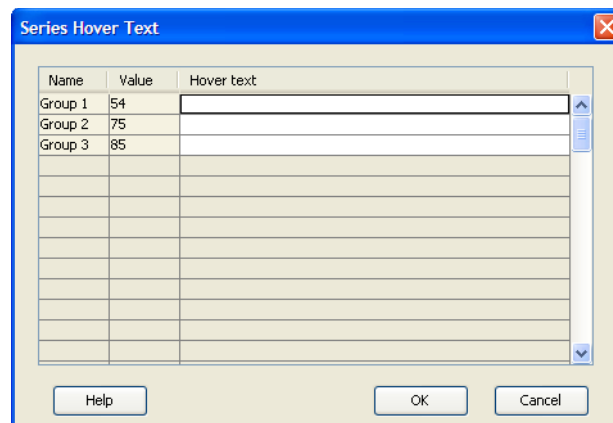
To add graph hover text

- 1 **Select the graph.**
- 2 **Locate the appropriate data series in `Object Properties`.**
- 3 **In the `Hover Text` property, click the ellipsis (...) button to open the `Series Hover Text` dialog.**

All meta tags supported in the general `Hover Text` property can also be used to create series-level hover text.

- 4 **Enter the desired hover text in the `Hover Text` field for the appropriate data item(s).**

The `Name` and `Value` columns indicate the category to which each Hover Text field belongs.



Note: When adding text, create a line break with a `
` tag or a backslash n (`\n`). When the hover text is displayed, the line wraps at that point.

Note: As an alternative to creating hover text for each data item, consider creating a hover object with dynamic label text for each data item that requires it. For more information on this functionality, see [“Connecting to Data Files,”](#) in the *Corda 7 Developer Reference*.

DYNAMIC HOVER TEXT

Using an Image Template file with Corda Server lets you add hover text dynamically based on live data. This dynamic hover text overrides hover text defined in Corda Builder. Enable hover text dynamically with ITXML or PCScript.

ITXML

To add dynamic hover text to a graph with ITXML, configure a `cit: hover-text` property with the appropriate `cit: graph-settings`. Adding hover text in ITXML must be done when sending the data to Corda Server. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

PCSCRIPT

To add dynamic hover text to a graph with PCScript, use the `graph.AddhoverText ()` method.

MAP HOVER TEXT

Map hover text can be applied at the following points from **Object Properties**:

- Map layer: **Hover Text** property in **Default Shape**.
- Range: **Hover Text** property in the appropriate **Range**.
- Map shape: **Hover Text** property in the appropriate **Map Point** or **Map Point**.

Note: Because map shapes are just special instances of shape objects, hover text for regular shape objects can be configured in much the same way.

MAP META TAGS

Maps support the following hover text meta tags:

<code>_%CODE</code>	The map data item code. This is one of three ways to identify a map data item.
<code>_%LONGNAME</code>	The map data item’s long name. This is one of three ways to identify a map data item.
<code>_%CNAME</code>	This meta tag is used in conjunction with a data file that provides the desired label text. For more information about using these data files, see “Comma-Separated Value Files” on page 2-10 of the <i>Corda 7 Developer Reference</i> .

- DATA ANNOTATIONS

- *Hover Text*
-
-

%_NAME	The name of the data item. This is one of three ways to identify a map data item.
%_NAMECLASS	The type of item for the map data item (e.g., city, state, province, or zip code).
%_RANGENAME	The name of the range in which this map data item falls.
%_PARENT	The name of the map to which this data item belongs.
%_VALUE	The value of the data item.
%_VALUENOFOR MAT	The value of the data item, minus any additional formatting.

For more information on meta tags, see [“Data Value Format” on page 10-3](#).

MAP HOVER TEXT LAYOUT

Corda Builder offers you several options for map hover text. To configure a map’s hover text layout, use the `Layout` property in `Hover Text`, which provides control over hover text position, margin, and maximum width.

Properties and attributes related to map data label layout are available from the `Hover Text > Layout` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

Note: *Because map shapes operate in the same way as regular shape objects, all these positioning options are available to shape objects as well.*

DYNAMIC HOVER TEXT

Using an Image Template file with Corda Server lets you add hover text dynamically based on the live data. This dynamic hover text overrides hover text defined in Corda Builder. Enable hover text dynamically with ITXML or PCScript.

ITXML

To add dynamic hover text to a map shape with ITXML, add a `hover-text` property to the appropriate `Point Shape` or `Map Shape`. Adding hover text in ITXML must be done when sending the data to Corda Server. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

PCSCRIPT

To add dynamic hover text to a map with PCScript, use the `map.SetHover()` method.

SHAPE HOVER TEXT

Because map shapes are just a special case of shape objects, shapes support hover text in the same way as map shapes.

Properties and attributes related to shape hover text position are available from the `Hover Text` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

Note: *The only way to see hover text is to preview the Corda image or look at the saved Corda image in a Web browser.*

DYNAMIC HOVER TEXT

Using an Image Template file with Corda Server lets you add hover text dynamically based on the live data. This dynamic hover text overrides hover text defined in Corda Builder. Enable hover text dynamically with ITXML or PCScript.

ITXML

To add dynamic hover text to a map shape with ITXML, add a `Hover Text` property to the appropriate [Map Point](#) or [Map Shape](#). Adding hover text in ITXML must be done when sending the data to Corda Server. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

PCSCRIPT

To add dynamic hover text to a map with PCScript, use the `shape.SetHover()` method.

VIEWING HOVER TEXT

Because hover text does not show in the Corda Builder interface, preview the Corda image to see hover text.

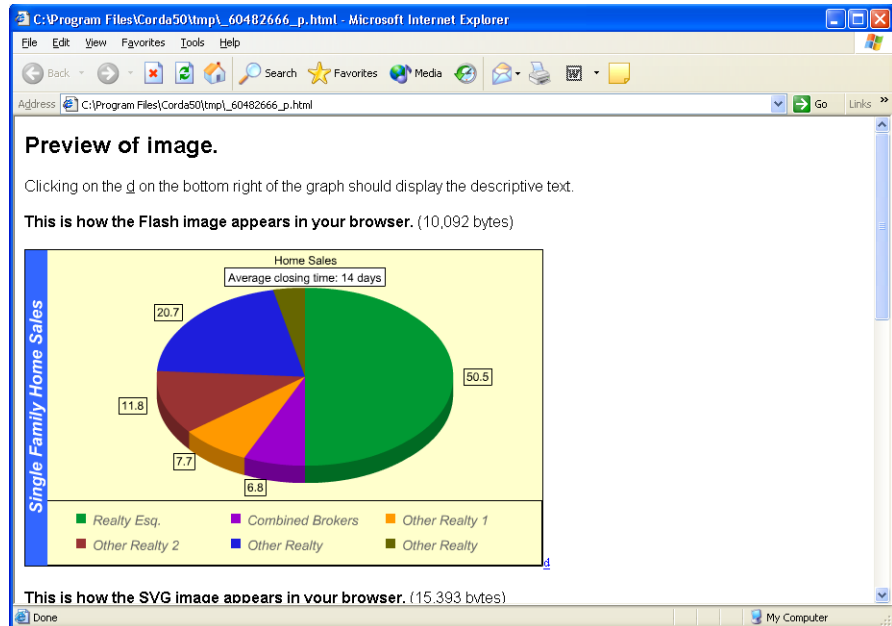
- DATA ANNOTATIONS

- *Hover Text*

To preview hover text

- 1 Select **File > Preview**, or click the  button on the toolbar.

The system's default Web browser opens and displays a Corda image similar to the one seen below.



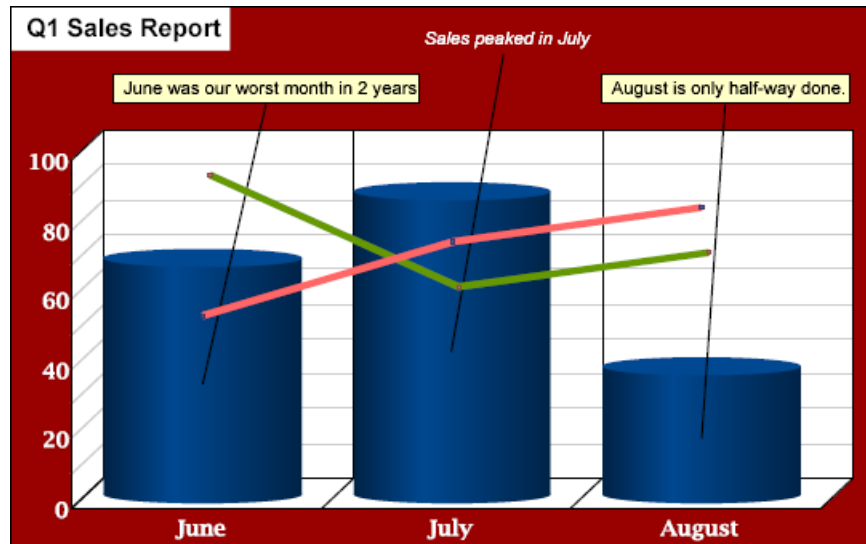
- 2 View the hover text by mousing over the Corda Image.

Warning: The Web browser determines how the hover text is displayed—a JPEG image might not be able display hover text at all on some browsers.

GRAPH NOTES

Graph Notes are a combination of text boxes and hover text. Another name for them is *Sticky Popups*. Like hover text, they are attached to a specific data item. However, like text boxes, they are always visible. Notes “call-out” from the data item—a leader line connects the data item to the note box.

Note: *All graph types, except for area graphs and gauges, support notes. Maps do not support notes.*



Properties and attributes related to graph notes are available from the `Note Text` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

All graph series (e.g., bar, pie wedge, or X-Y point) support notes.

To add a graph note

- 1 **Select the graph.**
- 2 **Locate the appropriate data series in `Object Properties`.**
- 3 **In the `Note Text` property, click the ellipsis [...] button to open the `Series Note Text` dialog.**
- 4 **Enter the desired note text in the `Note Text` field for the appropriate data item(s).**

- DATA ANNOTATIONS
- Graph Notes
-
-

The **Name** and **Value** columns indicate the category to which the **Note Text** field belongs.

Name	Value	Note text
Group 1	54	
Group 2	75	
Group 3	85	

Note: When adding text, create a line break with a `
` tag or a backslash `n` (`\n`). When the note is displayed, the line wraps at that point.

DYNAMIC NOTES

When using an Image Template file with Corda Server, you typically add graph notes dynamically based on the live data. These dynamic notes override those defined in Corda Builder. Enable notes dynamically with PCScript or ITXML.

ITXML

To use ITXML to add a dynamic note to a data item, set the `cit:note` property in the appropriate object property. Adding a note in ITXML must be done when sending the data to Corda Server. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

PCSCRIPT

To add a note dynamically in PCScript, use the `graph.AddNote()` method.

GENERAL GRAPH NOTE SETTINGS

Notes are displayed in text boxes. Once Note text is entered, a new Text box object is added to the project in which you can configure the look and feel of the Note. The only difference between a normal text box and a Note is the leader line from the Note text box to the data item.

Configure a Note text box just as you would any other text box object, by selecting it from the Project Canvas or Object List and modifying its properties in Object Properties. For more information on text boxes, see ["Text Boxes" on page 8-5](#).

Properties and attributes related to Note text box settings are available from the `Textbox Settings` property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

GRAPH NOTE LEADER LINES

Leader lines connect a graph note to its associated data item. Specify leader line color and thickness using the `Note Settings > Layout` property. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

- **DATA ANNOTATIONS**
- *Graph Notes*
-
-



ADDITIONAL INFORMATION

This chapter discusses some additional features related to Corda® Builder™ Image Template files. Topics in this chapter include:

- [Hover Objects](#)
- [Descriptive Text](#)
- [Data Tables](#)
- [User Preferences](#)
- [Numeric Format Settings](#)
- [Multiple Graphs and Maps in the Same Project](#)
- [Creating Templates](#)
- [Animations](#)
- [Creating New Color Themes](#)
- [Dynamic Objects](#)
- [Corda Fonts](#)

- ADDITIONAL INFORMATION

- *Hover Objects*

HOVER OBJECTS

A powerful feature of Corda Builder is the ability to create *hover* objects—that is, graph, map, image, shape, and other objects that appear as the user mouses over shape objects in the Corda Image.

Important: *Hover objects only work in SVG and Flash outputs.*

Hover objects are always visible in Corda Builder, but when previewing the image, you see the hover object only when mousing over a shape that you have selected to “trigger” the hover object.

Note: *This feature applies to map data items as well.*

Creating a hover object occurs in two steps. First, you need to define an object as a hover object. Second, you need to define a trigger shape—that is, a shape that makes the hover object appear.

In the example instructions provided in this section, **Shape 1** is a shape that appears constantly and acts as the trigger shape. **Shape 2** is the hover object. Note that **Shape 2** could just as easily be a graph, map, or a textbox. When a user moves his or her mouse over **Shape 1**, **Shape 2** appears.

- [Defining a Hover Object](#)
- [Defining a Trigger Shape](#)
- [Viewing a Hover Object](#)

DEFINING A HOVER OBJECT

Define an object as a hover object using the **Common > Is Hover Item** attribute in **Object Properties**. More information on this property is available in the [Corda 7 Object Reference](#).

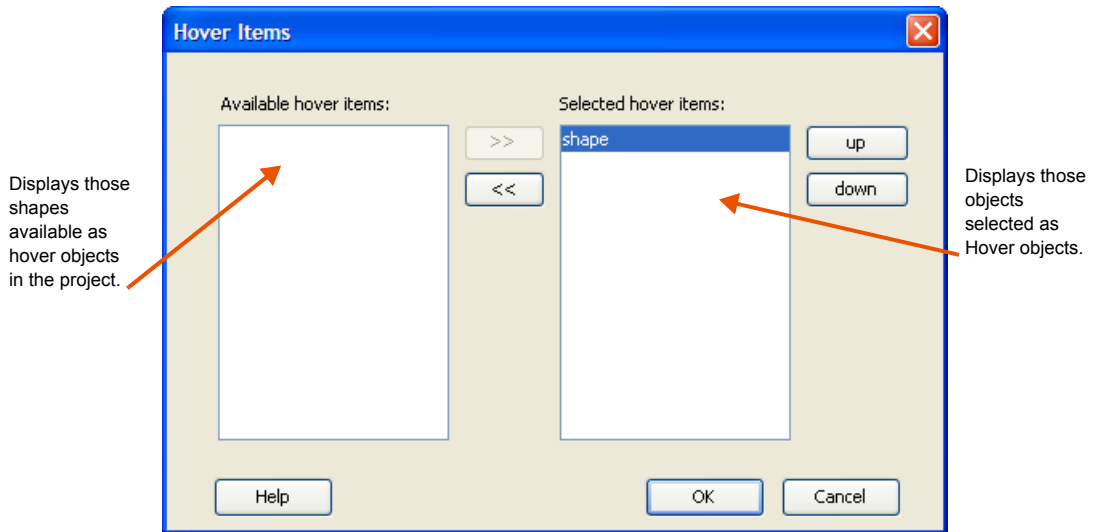
DEFINING A TRIGGER SHAPE

The second part of defining a hover object is to define a *trigger* shape—that is, a shape that, as a user moves his or her mouse over it, makes the hover object appear.

Any shape object, including map shapes and layers, can be used as a trigger shape. A trigger shape can make multiple hover objects appear. Likewise, a hover object can have multiple triggers.

Important: *The trigger shape must be in the foreground of the project. If any other shape or image is placed in front of it, the trigger shape is inaccessible in SVG output. To put a shape in the foreground, select it and then select **To Front** from the **Alignment Tools**.*

To define a trigger shape, specify the associated hover objects in the **Hover Items** property. Clicking the ellipsis [...] button opens the **Hover Items** dialog, which lets you select defined hover objects to associate to this trigger shape.




Note: Create hover objects for an entire map layer by defining the hover objects in the **Hover Items** property in **Default Shape**. For more information on a map's default shape properties, see ["Default Layer Properties" on page 6-8](#).

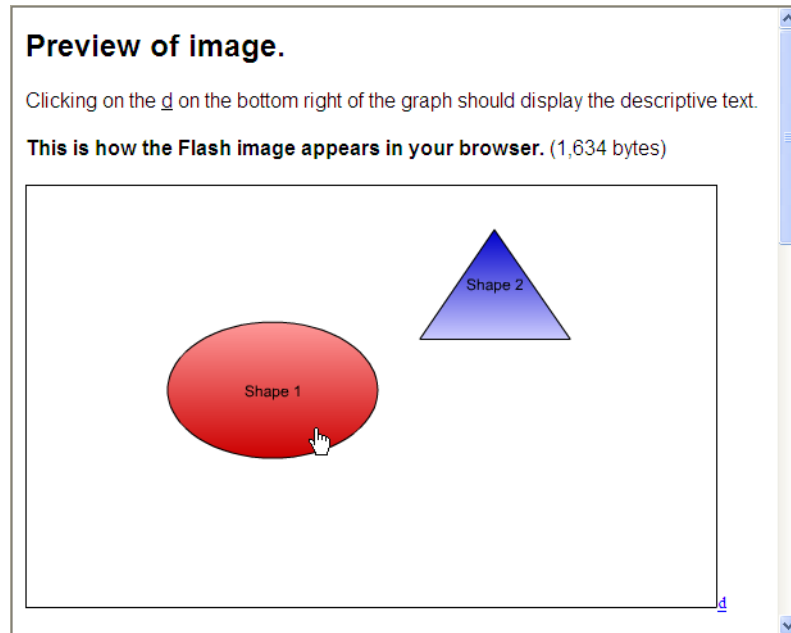
- **ADDITIONAL INFORMATION**
- *Hover Objects*
-
-

VIEWING A HOVER OBJECT

To see hover objects as they look in the published Corda Image, use Corda Builder's preview feature.

To preview a hover object

- 1 **Select File > Preview, or click the Preview  button on the toolbar.**
The system's default Web browser opens and displays the Corda image.
- 2 **View the hover object by mousing over the trigger shape.**



DESCRIPTIVE TEXT

Corda Technologies recognizes the importance of providing equal access to information for users who are visually impaired, and is the first and only company to provide multiple data visualization solutions that create accessible, 508-compliant graphs and maps with descriptive text.

There are an estimated 800,000 visually impaired persons currently using the Web. It is estimated that there are 6.5 million Americans age 55 or older that experience severe vision loss; by 2030 this number is projected to double.

The descriptive text is available for a screen reader device to audibly describe a graph or map so a visually impaired user can understand the graphical information. Corda Technologies has included the descriptive text feature in all members of its Corda Server™ family of data visualization solutions to facilitate inclusion of 508-compliant graphs and maps in all Web content.

Corda Server graph or map output is interactive for all users, including the visually impaired. Interactive graphs and maps enable all users to drill down to more in-depth information. Sighted users receive additional information by drilling down to another graph, map, or Web page. Non-sighted users access the same additional information by going from one chart with descriptive text to another chart with descriptive text. As the data changes, the graph or map and the descriptive text automatically change to match the new data. This fulfills the requirements of Section 508 of the Rehabilitation Act, which requires that all graphs and maps must be simultaneously offered in a text format that can easily be interpreted by a screen-reader such as JAWS and IBM* Home Page Reader.

Although this law applies only to federal organizations, many private organizations are also realizing the need to provide a method for visually impaired users to access important data.

This section includes the following topics:

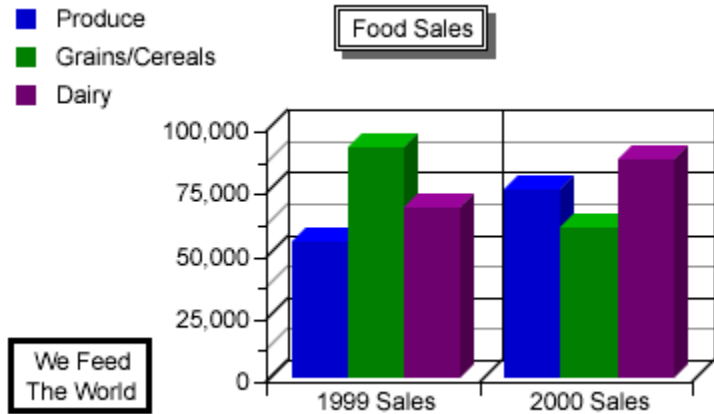
- [Descriptive Text Overview](#)
- [Enabling Descriptive Text](#)
- [Configuring Descriptive Text](#)

- ADDITIONAL INFORMATION
- *Descriptive Text*
-
-

DESCRIPTIVE TEXT OVERVIEW

When you enable descriptive text, a link appears either next to the bottom-right corner of the Corda image or below it. By default, this link is a blue underlined letter *d* ([d](#)). Selecting it takes the user to a Web page containing a textual description of the chart.

As a screen reader reads over this link, it indicates to the user that descriptive text is available for the image. The user then has the option of having the screen reader read the description of the Corda image.



[d](#)



Selecting the [d](#) takes you to a page similar to the following:

Example 11.1

Descriptive Text Example

Food Sales

Title of Corda image

Each drilldown item appears with the link right next to it, so users are still able to drill down.

Bar chart with 2 groups with 3 items per group.

Group 1, 1999 Sales

Item 1, Produce 54000, [Detail for 1999 Sales Produce](#)

Item 2, Grains/Cereals 92000, [Detail for 1999 Sales Grains/Cereals](#)

Item 3, Dairy 68000, Milk is Good!, [Detail for 1999 Sales Dairy](#)

Group 2, 2000 Sales

Item 1, Produce 75000, [Detail for 2000 Sales Produce](#)

Item 2, Grains/Cereals 60000, [Detail for 2000 Sales Grains/Cereals](#)

Item 3, Dairy 87000, Milk is Good!, [Detail for 2000 Sales Dairy](#)

Text boxes are listed at the end.

We Feed The World

[back](#)

Users can go back to the page they came from after they are done reading the descriptive text.

Descriptive text indicates items with hover text.

This data is fully interactive. What would appear as hover text in a graphical Corda image appears as text right beside the description for each item. Likewise, drilldown links appear next to items that are enabled for drilldown effects. Visually impaired users are able to drill down from this descriptive text to another Corda image with [d](#) link descriptive text.

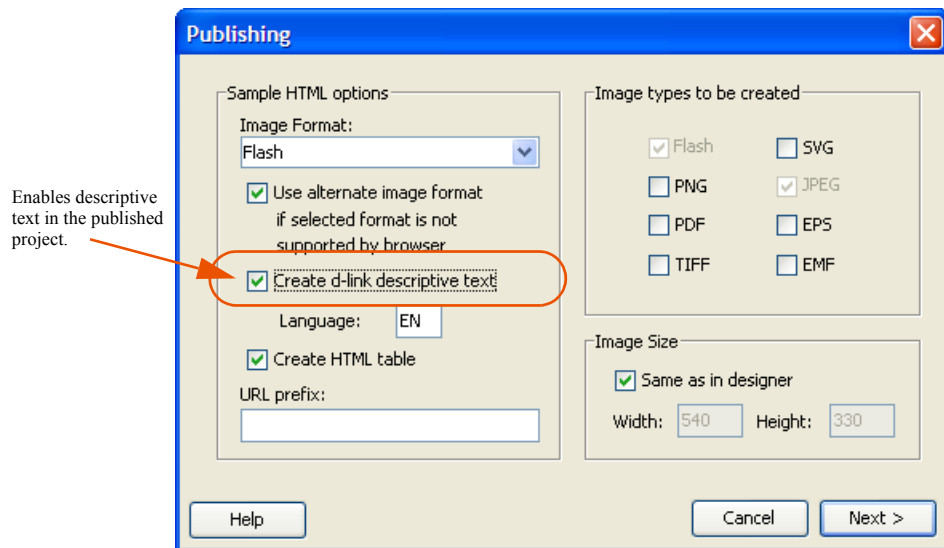
Note: *At this point in time, drilldown to custom Javascript functions or named destinations within the original Web page is not supported in the **Corda Builder** descriptive text output. You may be able to circumvent the Javascript limitation, however, by customizing the descriptive text template so that it includes a predefined Javascript library of the functions you need to access.*

- ADDITIONAL INFORMATION

- *Descriptive Text*

ENABLING DESCRIPTIVE TEXT

When publishing a static graph or map with Corda Builder, select **Create d-link descriptive text** when you publish the Corda Image. For more information on publishing Corda Images, see Chapter 13, “[Publishing the Project.](#)”



When publishing an Image Template file for use with dynamic data from Corda Server, enable descriptive text in the code used to embed the Image Template file in the Web application. For more information, see Chapter 8, “[Image Deployment Issues,](#)” of the *Corda 7 Developer Reference*.

CONFIGURING DESCRIPTIVE TEXT

Configure descriptive settings in the `DLink Settings` property. More information on this property is available in the [Corda 7 Object Reference](#).

Corda Builder lets you manage the specifics of descriptive text output in the following ways:

- [Descriptive Text Title](#)
- [Custom Descriptive Text](#)
- [Descriptive Text Overrides](#)
- [Modifying the Descriptive Text Template](#)

DESCRIPTIVE TEXT TITLE

When Corda Builder (or Corda Server) generates descriptive text, it uses the contents of the Title text box as the title at the top of the descriptive text page. Modifying the Title text box in a Corda Image changes the descriptive text title.

Note: *If a Title box is defined for a Corda Image, it is always displayed as part of the descriptive text.*

For more information on the Titles and Text boxes, see [“Text Boxes” on page 8-5](#).

CUSTOM DESCRIPTIVE TEXT

The **Description Type** attribute in the **DLink Settings** property lets you configure descriptive text for automatic generation or manual generation.

automatic Uses the descriptive text template to automatically generate descriptive text content. Use the **Overrides** attribute to add or remove specific Corda Image content from descriptive text. For more information, see [“Descriptive Text Overrides” on page 11-9](#).

manual Disables automatic descriptive text generation and relies solely on the content you specify in the **Description** attribute, and the contents of the Title box, if one is defined.

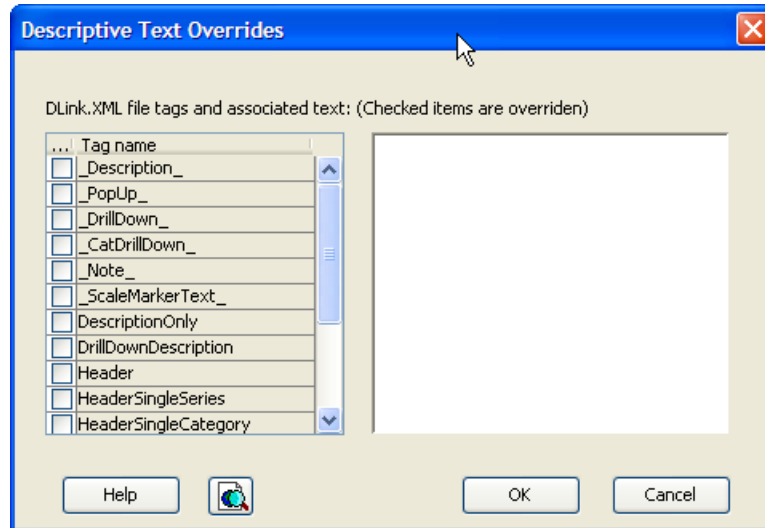
DESCRIPTIVE TEXT OVERRIDES

Corda Builder gives you complete control over the objects in a project described by descriptive text, as well as how those objects are described. A descriptive text template file defines these settings for an object, but **Object Properties** let you override these template settings on a object-by-object basis.

- ADDITIONAL INFORMATION
- *Descriptive Text*
-
-

To override descriptive text template settings

- 1 In **Object Properties**, open the **DLink Settings** property.
- 2 Click the ellipsis [...] button in the **Overrides** property to open the **Descriptive Text Overrides** dialog.



- 3 Check the boxes next to those dlink tags to override the settings, and click **OK**.

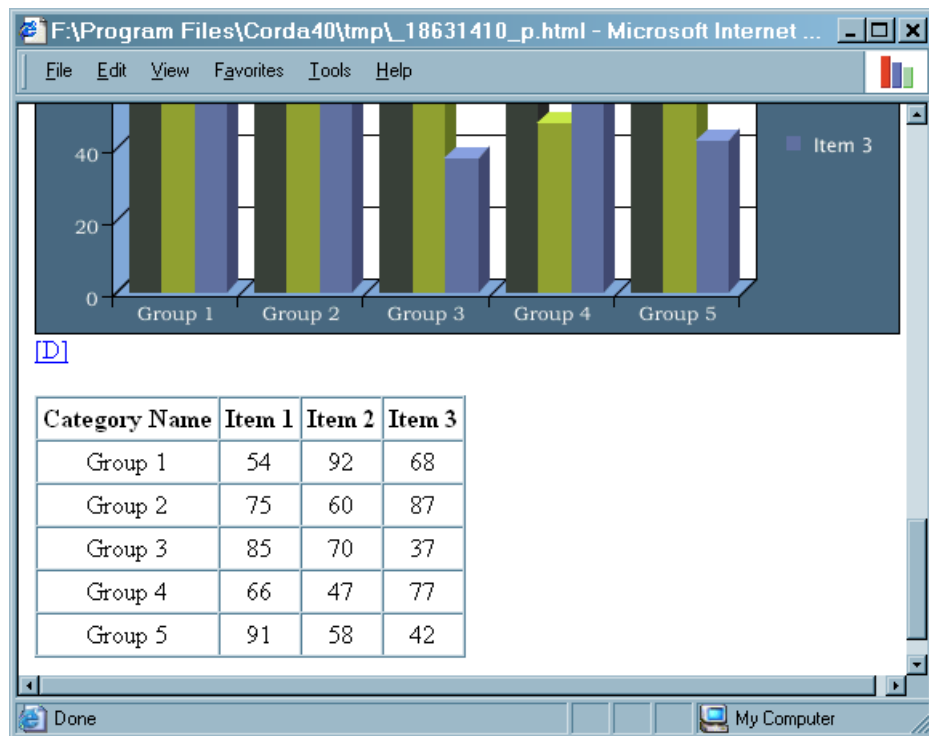
Click the **Preview** button (next to the **Help** button) to preview the descriptive text changes.

MODIFYING THE DESCRIPTIVE TEXT TEMPLATE

From the standpoint of a novice user, modifying the descriptive text template might be the most difficult method of changing the descriptive text format because of the high learning curve involved with understanding how the template works. For advanced users, however, this is a powerful and convenient way to control descriptive text. For more information, see Chapter 10, “[Descriptive Text Settings](#),” in the *Corda 7 Developer Reference*.

DATA TABLES

Corda Server can automatically display an HTML data table below a graph or map. The HTML table displays the data used in the graph or map. Optionally, the data table can contain an extra column and row for data summation.



To enable data tables

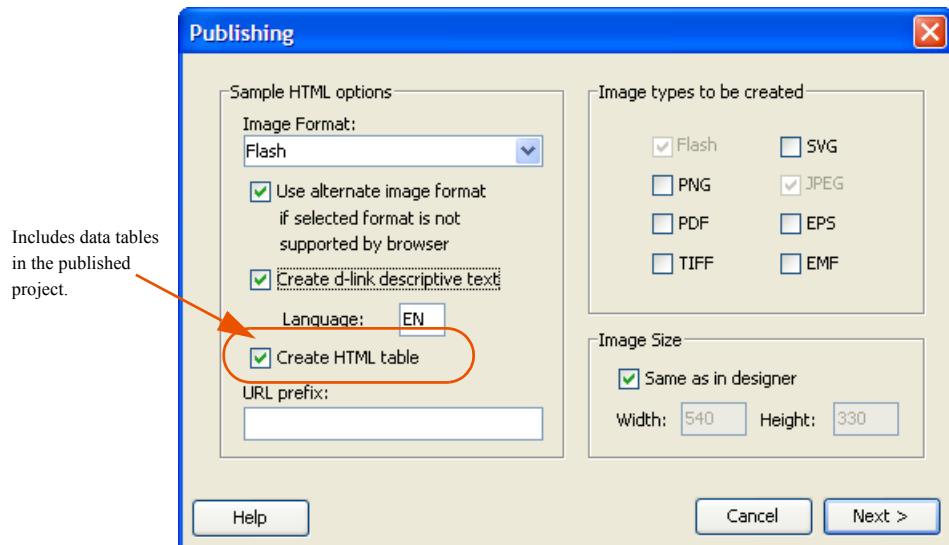
When publishing an Image Template file to Corda Server for dynamic graph and map display, enable data tables in the code that you use to embed the Image Template file in Web applications. For more information, see Chapter 8, “[Image Deployment Issues](#),” in the *Corda 7 Developer Reference*.

When publishing a static graph or map with Corda Builder, select **Create d-link descriptive text** when you publish the Corda Image. For more information on publishing Corda Images, see Chapter 13, “[Publishing the Project](#).”

- **ADDITIONAL INFORMATION**

- *Data Tables*

If you're publishing a static version of the graph or map with Corda Builder, enable data tables when you publish an image by checking the **Create HTML table** box.



DATA TABLE STYLES

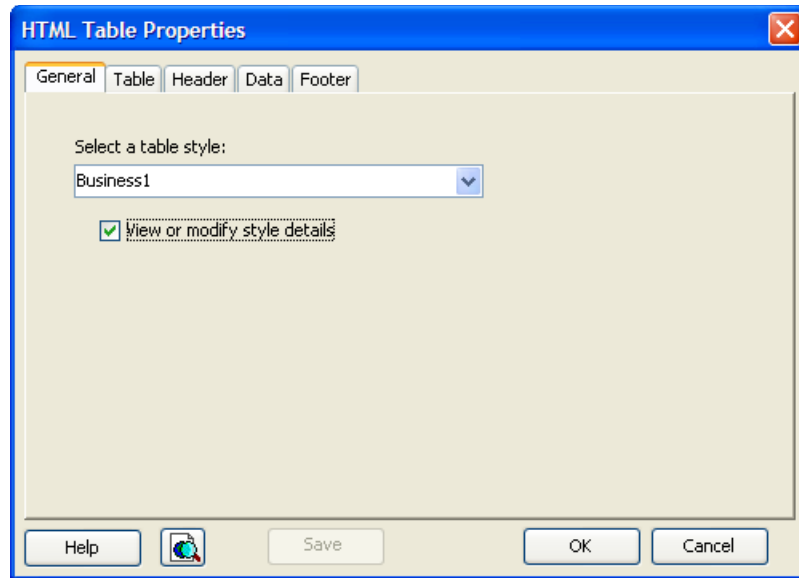
Corda Builder provides a wide variety of pre-built styles for customizing the HTML data table's look and feel. The style specifies formatting and attributes such as cell colors, fonts, and even layout. Each graph and map in an Image Template file specifies its own style.

Properties and attributes related to data table styles are available from the **HTML Table** property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

CREATING OR MODIFYING DATA TABLE THEMES

When specifying a data table style, click the ellipsis [...] button to open the **HTML Table Properties** dialog. Select **View or modify style details** to enable four new tabs in the

dialog that allow you to modify the style's settings. Find out more about these tabs by clicking the [Help](#) button.



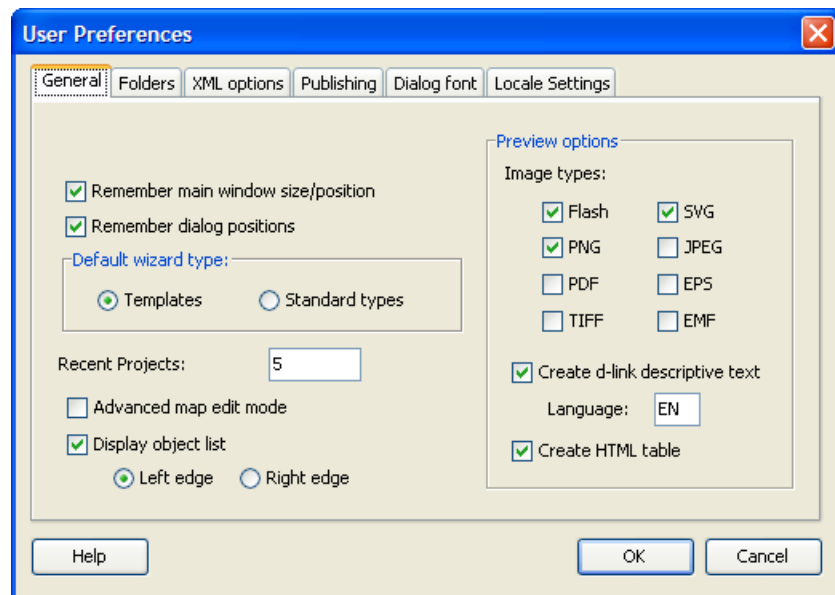
When you are done modifying the style, click the [Save](#) button to save the new table style to the template file.

• **ADDITIONAL INFORMATION**

• *User Preferences*

USER PREFERENCES

If you use Corda Builder frequently, you may find it useful to customize the way Corda Builder works. The **User Preferences** dialog lets you change the default save and template locations, specify how Corda Builder should start up, choose the default preview options (such as image format) for the projects, and select which font Corda Builder should use in its dialogs.



To change User Preferences

- 1 **Select *Edit > Preferences*.**
- 2 **Make the appropriate modifications in the *User Preferences* dialog.**
- 3 **Click *OK* to save the changes.**

User Preferences let you configure Corda Builder preferences using the following tabs:

- [General](#)
- [Folders](#)
- [XML Options](#)
- [Publishing](#)
- [Dialog Font](#)
- [Locale Settings](#)

GENERAL

Determines how Corda Builder behaves in general.

REMEMBER MAIN WINDOW SIZE/POSITION

Specifies that Corda Builder restarts at the same location and with the same size on the screen as when you last exited the program.

REMEMBER DIALOG POSITIONS

Specifies that Corda Builder places new dialogs in the same location as when they were last closed. If you move a dialog once, you don't have to do it again.

RECENT IMAGE TEMPLATES

Specifies the number of Image Template files that Corda Builder remembers in the **Files > Recent Image Templates** menu item.

ADVANCED MAP EDIT MODE

Enables map editing in advanced mode, meaning you can move and modify shapes within the map as if they are standalone shapes.

PREVIEW OPTIONS

Controls how Corda Builder shows Corda images when you click the Web browser's preview button.

Image Types Specifies the preview formats for a Corda image. Select the formats in which you want to preview the image. The following settings are available:

- Flash
- SVG*
- PNG
- JPEG
- PDF
- EPS
- TIFF
- EMF

Create d-link Descriptive Text Instructs Corda Builder to generate descriptive text for the visually impaired with each preview. The related Language field specifies the language used to generate the descriptive text.

- **ADDITIONAL INFORMATION**

- *User Preferences*
-
-

Create HTML Table Instructs Corda Builder to generate an HTML table with each preview.

FOLDERS

DEFAULT SAVE LOCATION

Specifies default save options for any Image Template files (projects) you create with Corda Builder. By default, this folder is the <document_root> folder.

Note: *This folder is different from the folder where published projects are stored (see [Publish Folder](#) below).*

MAP FOLDER

Specifies the location for maps. Corda Builder looks at this location when you create a map from a template file. By default, it is set to the <document_root>\maps folder inside the Corda Builder root directory.

PUBLISH FOLDER

Specifies the directory to which Corda Builder saves its published projects. By default, this folder is the <document_root>\publish folder of the Corda 7 installation. Corda Builder writes all image and HTML files it generates to this directory.

Note: *The Image Template file for a project is saved to a different location than the published images and Web page. The Image Template file save location is specified using [Default Save Location](#) at the top of the dialog.*

XML OPTIONS

Controls how Corda Builder saves Image Template files.

APPEARANCE PROPERTIES

Specifies how Corda Builder generates an Image Template file's ITXML.

Verbose Outputs all attributes and elements, regardless of whether or not they are using the default value. Not only does this setting produce a large Image Template file, but it takes Corda Server up to four times longer to process this Image Template file. Provided mostly to help you learn more about ITXML.

Minimal Removes any attributes or elements that are set to their default value, making an Image Template file as small as possible. This is the default setting.

DATA FORMATTING

Specifies how Corda Builder saves the data in an Image Template file. The only reason you want to change this setting is to learn how ITXML should look if you need to import data in ITXML using a row/column spreadsheet format instead of the default series/categories/data format.

Note: The following settings apply only to graphs.

Native Exports the data as it normally does using *Series*, *Category*, and *Data* tags. This is the default setting.

Row/Column Exports the data in spreadsheet form using *Row* and *Cell* tags.

SHOW ITXML ERRORS IN BUILDER

Specifies whether Corda Builder notifies you of ITXML document errors it detects. By default, this is turned off. This functionality is provided to help you check the validity of ITXML documents.

PUBLISHING

Specifies default publishing options.

REDUCE UI FOR PUBLISHING

Instructs Corda Builder to hide the *ITXML* tab at the bottom of the Corda Builder window. Use this option when you are only publishing static graphs and maps (as opposed to creating Image Template files and sample code for Corda Server™).

DEFAULT SAMPLE HTML OPTIONS

Specifies default presentation options for the sample Web page produced during the publish process.

Image Format Specifies the image format that is generated when you save the graph. Available formats include *Flash*, *SVG*, *PNG*, and *JPEG*.

Use Alternative Image Format (Fallback) Enables best-image fallback. This means that if a user's Web browser does not have the proper plug-in to display the Corda image in the format you have selected, the browser tries to display it in alternate formats. This allows you to take advantage of high-quality formats with the assurance that even if the end user cannot view these formats, they can still see the Corda image in the JPEG format.

Create d-link descriptive text Generates descriptive text for a Corda image in a format that can be read by a screen reader for the visually impaired. Additional information regarding descriptive text is discussed in ["Descriptive Text" on page 11-5](#).

- **ADDITIONAL INFORMATION**

- *User Preferences*

- **Create HTML Table** Generates an HTML table for a Corda image that contains the same data as the graphs and maps in an image. This table appears immediately below the image.

IMAGE TYPES TO BE CREATED

Specifies default options for the formats in which Corda Builder saves an image when it is published. Select the formats in which you want to save the image. Certain formats are automatically selected based on settings in [Default Sample HTML Options](#).

The following image formats are available:

- Flash
- SVG
- PNG
- JPEG
- PDF
- EPS
- TIFF
- EMF

JAVASCRIPT HOVER FOR PNG AND JPEG

Instructs PNG and JPEG images to use an image map to provide hover text and drilldown effects rather than Javascript.

Warning: Hover and drilldown capabilities of image maps are greatly limited in comparison to those provided by Javascript.

DIALOG FONT

Changes the system font used by Corda Builder. If you are unable to see international or double-byte characters when you type them in text boxes, you can resolve that problem by choosing a font that supports international or double-byte characters. To change the font, select it by double-clicking or pressing **Select**; then click **OK**.

You can change the font back to the default font by clicking the **Use Default** button.

LOCALE SETTINGS

Specifies default preferences suitable for a selected country/language combination. Locale settings include such things as Number format, Percentage format, Currency format, and Date format. Changing the Locale setting changes the default configuration of these formats.

By default, **Corda Builder** selects a default locale based upon operating system settings. To change the settings **Corda Builder** uses, select not to use the default locale and choose another locale from the list of supported locales.

Selecting [Advanced Locale Configuration](#) lets you configure the settings for the selected Locale.



- **ADDITIONAL INFORMATION**

- *Numeric Format Settings*
-
-

NUMERIC FORMAT SETTINGS

Corda Builder lets you customize several properties that define numeric formats at the project level, including the following:

DECIMAL SEPARATOR Specifies the decimal indicator used in the project (for example, the period in 4.83). Supported values include **Period**, **Comma**, and **Space**.

THOUSANDS SEPARATOR Specifies the numeric separator that separates every three digits (for example, the commas used in 1,000,000). Supported values include **Period**, **Comma**, and **Space**.

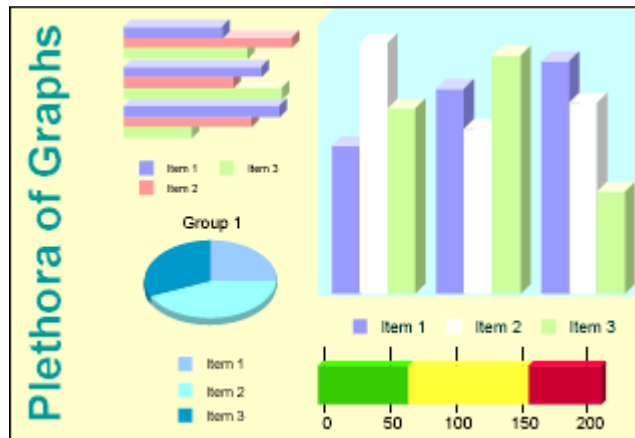
CURRENCY SYMBOL Specifies the currency symbol that is inserted when a scale label is specified as type **Currency** (for example, US \$). It also lets you specify the position of the currency symbol (before or after the currency value).

NUMBER ABBREVIATIONS Specifies the abbreviation to use for different numbers when abbreviations are enabled in a project. For example, setting the abbreviation for Billion to “B”; then 68,970,000,000 displays as 68.97B.

Properties and attributes related to numeric format settings are available from the **Number Format** property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

MULTIPLE GRAPHS AND MAPS IN THE SAME PROJECT

One very useful feature of Corda Builder is the ability to place multiple graphs and maps in the same project. Each graph or map can have its own data, legend, and settings.



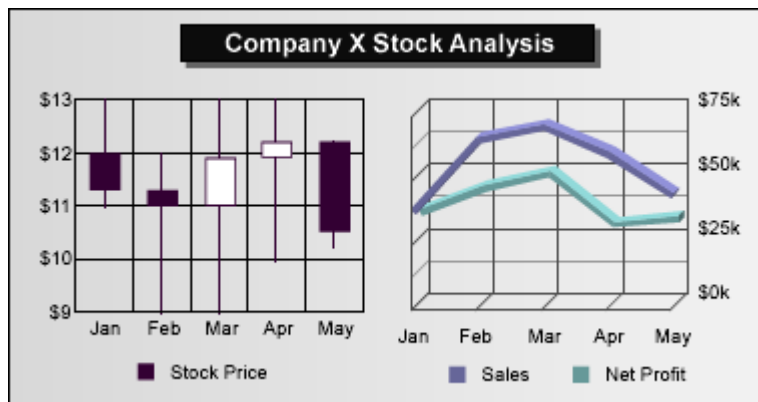
When using an Image Template file with Corda Server as a dynamically generated Corda image, be aware of each object's name. You must specify an object's name when dynamically changing its data or formatting.

By default, Corda Builder gives each object in a project a unique name. Object names are stored in, and can be changed from, the `name` property. More information on this property is available in the [Corda 7 Object Reference](#).

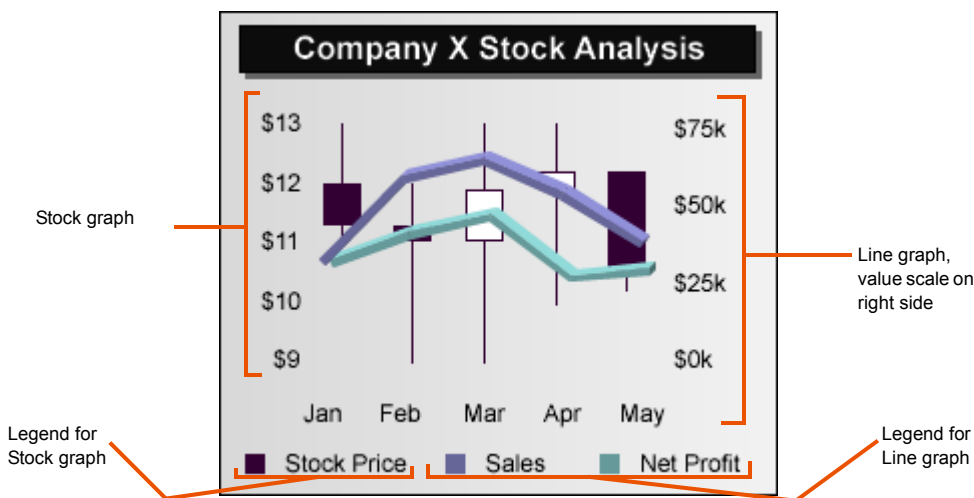
- **ADDITIONAL INFORMATION**
- *Multiple Graphs and Maps in the Same Project*
-
-

OVERLAYING GRAPHS

One reason to have multiple graphs in a project is to overlay one graph over another. This is useful for comparing two different data sets with different graph types and/or with different data ranges. Consider the following:Corda image:



These images might be more interesting and useful by overlaying the Line graph on top of the Stock graph, so readers can more easily make a visual comparison of the data. The next Corda image shows what this might look like.



ADDITIONAL INFORMATION*Multiple Graphs and Maps in the Same Project*

Overlaying graphs sometimes requires modifications to grid lines and scales to achieve this effect. In the example above, we removed the bottom scale of the Stock graph, so that the two bottom scales didn't overlap. Because of this, we also had to change the size of the Stock graph. Also note that we put the scale for the Line graph on the right side of the graph, so that the two graph scales don't overlap. For more information on working with scales and grid lines, see Chapter 5, [“Scales and Grids.”](#)

The items on the [Alignment](#) menu of the Corda Builder interface are helpful in overlaying graphs, especially the [Move to Front](#) and [Move to Back](#) items (see [“Aligning Objects”](#) on page 3-4).

- **ADDITIONAL INFORMATION**

- *Creating Templates*
-
-

CREATING TEMPLATES

If you or the company create many graphs or maps that are similar to each other, it may be beneficial to create a Corda Builder template. Because the template is already customized to suit a company's needs, the only thing you have to do to create the final Corda image is select this template and enter in the new data. This greatly simplifies the Corda Builder design process.

To create a Corda Builder template

- 1 **Create a project that looks exactly like you want the template to look.**

For information about formatting a template, see ["Tips for Creating Graph Templates."](#)

- 2 **Select File > Save Canvas as Gallery Item.**

This launches the Gallery dialog, from which you can specify a storage location and name for the project. For more information about the Gallery, see ["The Corda 7 Gallery" on page 3-8.](#)

TIPS FOR CREATING GRAPH TEMPLATES

The following are tips for creating graph templates:

- Make the project self-explanatory, so users know what they should modify and what they should leave alone. For example: a text box where users enter today's date should include a title *Enter Today's Date* for the text box.
- Decide on the image format and size that projects created from the template need to be set to. Set these in the template using [Project Properties](#).
- It is difficult to anticipate what or how much data is put into the template. A graph that looks good with a very small set of sample data may look horrible with twenty or thirty categories and series. Test the template with a variety of data to make sure that it still looks good.
- Avoid manually setting scales. Doing so could inadvertently exclude data from live data sets.
- Decide how you want the graph to automatically adjust the bottom scale labels. For more information, refer to ["Automatic Label Adjustments" on page 5-18.](#)
- Again, because the data may vary, be careful when setting drilldown URLs or hover text. Only set these when you are certain that the data values you set always use the drilldown or hover settings.

ANIMATIONS

Corda Builder supports two types of animations:

- [Map Fade-Ins](#)
- [Graph Animations](#)

These animations only work in Flash images.

Warning: Animations are not compatible with automatically updated Flash images.

MAP FADE-INS

For maps, Corda Builder supports a fade-in effect, meaning the map fades gradually into view over the course of a few seconds.

To specify that a map should fade in, select [Map Settings > Animate \(Flash\)](#) from [Object Properties](#). More information on this attribute is available in the [Corda 7 Object Reference](#).

GRAPH ANIMATIONS

Corda Builder can also animate graphs containing any type of line, bar, or pie component. Bars appear to grow out of the bottom scale. Lines start out completely horizontal and then gradually morph into their true forms. Pie wedges fly into the graph from the outside edges of an Image Template file.

To animate a graph, select [Graph Settings > Animate \(Flash\)](#) from [Object Properties](#). More information on this attribute is available in the [Corda 7 Object Reference](#).

- ADDITIONAL INFORMATION

- *Creating New Color Themes*
-
-

CREATING NEW COLOR THEMES

Corda graphs use color themes to define a set of colors it applies, in succession, to graph series. For example, the first series gets the first color in the theme, the second series gets the second color in the theme, and so forth.

Corda Builder comes with several predefined color themes, but you may find it useful to develop custom color themes to fit your organization's specific needs. Add a new color theme by modifying `PCColors.xml`.

Important: *Back up any files that you want to modify before actually changing them.*

To create a new color theme

- 1 **Locate** `<product_root>\Resources\config\PCColors.xml`.
- 2 **Open the file using an XML or text editor.**

The file looks something like this:

Example 11.2 Color Themes XML File

```
<PopChartColorSchemes Version="1.0">
  <Scheme Name="Default">
    <Color1 Red="0" Green="153" Blue="51" />
    <Color2 Red="153" Green="0" Blue="204" />
    <Color3 Red="255" Green="153" Blue="0" />
    <Color4 Red="153" Green="51" Blue="51" />
    <Color5 Red="30" Green="30" Blue="220" />
    <Color6 Red="102" Green="102" Blue="0" />
    <Color7 Red="255" Green="102" Blue="0" />
    <Color8 Red="0" Green="102" Blue="204" />
    <Color9 Red="204" Green="204" Blue="0" />
    <Color10 Red="0" Green="110" Blue="0" />
    <Color11 Red="102" Green="153" Blue="255" />
    <Color12 Red="200" Green="30" Blue="30" />
    <Color13 Red="204" Green="153" Blue="255" />
    <Color14 Red="135" Green="84" Blue="0" />
    <Color15 Red="220" Green="10" Blue="130" />
    <Color16 Red="0" Green="85" Blue="0" />
  </Scheme>
  ...
</PopChartColorSchemes>
```

- 3 **Add a new color scheme by adding another `scheme` property to `PCColors.xml`.**

ADDITIONAL INFORMATION

Creating New Color Themes

The easiest way to do this is to copy an existing `scheme`, paste it at the end of `PCColors.xml`, and then modify the attributes and child properties as needed.

Each `Scheme` property includes a `Name` attribute, which specifies the scheme name, and sixteen `color` child properties, `Color1` through `Color16`, each of which represents one color in the color scheme. The `color` properties have the following attributes:

- **Red:** Specifies the R value in the RGB setting; can be any whole number between 0 and 255.
- **Green:** Specifies the G value in the RGB setting; can be any whole number between 0 and 255.
- **Blue:** Specifies the B value in the RGB setting; can be any whole number between 0 and 255.
- **Alpha:** Optional attribute. Specifies the transparency of the color; can be any whole number between 0 and 255. An `Alpha` of 0 makes the color solid, and an `Alpha` of 255 makes the color invisible, with varying degrees of translucency in between these extremes.

For example:

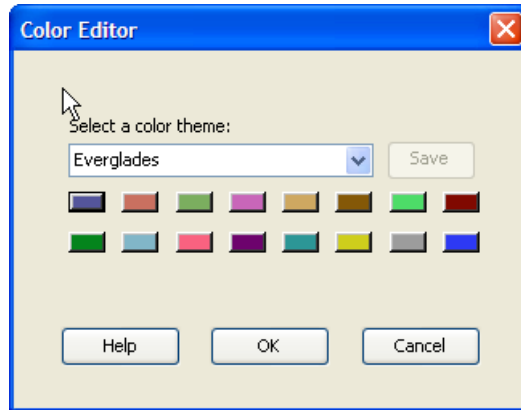
```
<Color1 Red="255" Green="0" Blue="0" Alpha="80"/>
```

4 Save the file.**5 Restart Corda Builder and/or the Corda Server.**

To modify existing themes in the `PCColors.xml` file, change the `Red`, `Green`, and `Blue` attributes for the appropriate `<ColorXX>` tags in the desired `<Scheme>` property. Alternately, click the ellipsis button [...] in **Graph Settings > Color Palette** to open the

- **ADDITIONAL INFORMATION**
- *Creating New Color Themes*
-
-

Color Editor, from which you can modify specific color settings in an existing color theme.



DYNAMIC OBJECTS

Dynamic objects are “placeholder” objects that can be dynamically replaced with other objects when the project is published as an image. They are usually only useful when you are using an Image Template file with Corda Server.

Dynamic objects define basic style definitions that are applied to the object dynamically inserted by Corda Server. This lets you maintain a consistent look and feel across the objects in an Image Template file.

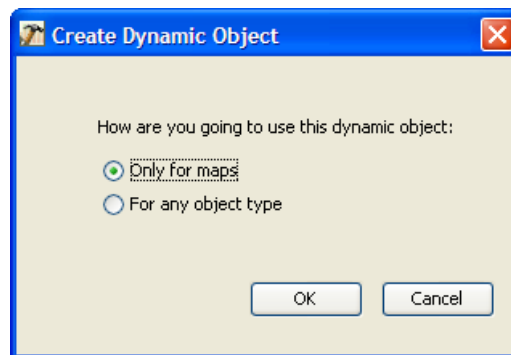
Dynamic objects are especially useful for dynamically swapping maps into an Image Template file, so that you do not have to create a new Image Template file for each map you want to display.

CREATING DYNAMIC OBJECTS

Corda 7 supports two different types of dynamic objects, one for maps and one for objects of other types. Map dynamic objects include additional configuration options that are not applicable to other object types.

To add a dynamic object

- 1 Select **Insert > Insert Dynamic Object**.
- 2 In the **Create Dynamic Object** dialog, specify if the Dynamic Object is for maps only or for objects other than maps, and click **OK**.



- **ADDITIONAL INFORMATION**

- *Dynamic Objects*

Corda Builder adds an empty frame (the Dynamic Object) containing the words *Dynamic Map Object* or *Dynamic Object* to the project. Resize, copy, and delete this object just as you do any other object in the project.



CONFIGURING DYNAMIC OBJECTS

Once created, configure dynamic objects in [Object Properties](#) as you would any other object in a Corda Builder project.

PREVIEW

When previewing an Image Template file, you might want to see an actual object of the type that will replace the dynamic object. Specify a preview object for the dynamic object from [Object Properties](#). Doing so involves two attributes:

Preview File Name Specifies the full name, including the path, of the Image Template file that contains the desired object.

Preview Object Name Specifies the name of the specific object within the Image Template file. The name of the object is defined in the [Name](#) attribute.

More information on these attributes is available in the [Corda 7 Object Reference](#).

PCSCRIPT

Corda Builder automatically generates basic PCScript for objects in an Image Template file, including dynamic objects. The Corda Builder-generated PCScript focuses on those objects that are most likely to be dynamically updated by Corda Server. This default PCScript is useful as a template for creating custom PCScript you might need to manipulate dynamic objects with Corda Server.

View default PCScript from either the [PCScript](#) or [Sample Code](#) dialogs:

PCScript Dialog Click the [PCScript](#) button in the toolbar. In the [PCScript](#) dialog, click [Insert Default PCScript](#).

Sample Code Click the [PCScript](#) button in the toolbar. The default PCScript is included as part of the sample code generated in the [Sample Code](#) dialog.

MAP LAYERS

Map dynamic objects automatically include two layers, one [Area](#) layer and one [Points](#) layer. These dynamic object layers let you configure the look and feel applied to the layers of the dynamically provided map that replaces the dynamic object.

When multiple layers of the same type are in the dynamically provided map, layer settings from the dynamic object are applied in the following ways:

Area layer Dynamic object settings apply to the backmost Area layer (the Area layer created first).

Points layer Dynamic object settings apply to the top Points layer (the Points layer created last).

For information about replacing a dynamic object in an Image Template file, see [“Dynamic Objects” on page 4-16](#) of the *Corda 7 Developer Reference*.

- **ADDITIONAL INFORMATION**

- *Corda Fonts*

CORDA FONTS

Corda Builder uses a proprietary font format (the Corda font format) that allows it to quickly generate many different types of images using a variety of character sets. This font format is proprietary for two reasons: 1) to avoid licensing issues, and 2) to keep Corda fonts system-independent so they work the same on any operating system. This font format provides full support for double-byte characters.

Because of the wide variety of fonts available, Corda Builder comes installed with only a small number of commonly used fonts.

However, if you have a font you want to use with Corda Builder, you might be able to convert that font to the Corda font format, as explained in [“Creating Custom Fonts.”](#)

Corda fonts are stored in `<product_root>\Resources\lib\fsfiles`. Fonts use an `.fsd` extension. Corda fonts are shared by both Corda Server and Corda Builder.

This section covers the following topics:

- [Creating Custom Fonts](#)
- [Installing Custom Fonts](#)
- [Using Custom Fonts](#)
- [International Fonts and Character Support](#)

CREATING CUSTOM FONTS

Import *TrueType* Fonts* into Corda Builder using the Corda Font Converter. This utility installs automatically with Corda Builder.

Corda Font Converter can convert fonts that meet these requirements:

- The font must be a Windows TrueType font.
- The font must have one of the following character mappings:
 - Unicode
 - Shift JIS (MS932) (for Japanese)
 - Big5 (MS950) (for Traditional Chinese)
 - GBK (aka PRC) (MS936) (for Simplified Chinese)
 - Wansung (MS949) (for Korean)
- The font must not create characters by combining multiple glyphs from the font. (The conversion completes successfully, but the characters do not display correctly.)

Because of the TrueType font requirement, it is easiest to run this program on Microsoft Windows systems, which use TrueType fonts natively. If you need to import fonts on another platform, either convert them on a Windows machine and install them following the directions in “Installing Custom Fonts” on page 11-34, or copy the necessary TrueType font files (with a .ttf extension) to the system on which you need to import the fonts.

CONVERTING TRUETYPE FONTS WITH THE CORDA FONT CONVERTER

Converting fonts to the Corda font format is a straightforward process.

To use the Corda Font Converter

1 Start Corda Font Converter.

Launch Corda Font Converter by selecting **File > Font Converter** in Corda Builder.

2 In the **Fonts field, enter the path to the TrueType font to convert, or click **Browse** (to the right of the **Fonts** field) to locate the directory that contains the font.**

When converting a system font—for example, fonts that you already use in other programs (like Microsoft Word)—look for the font in %SYSTEM%\Fonts, where %SYSTEM% is the directory where Microsoft Windows is installed (e.g., C:\Winnt or C:\Windows).

If this is the first time running the Corda Font Converter, the location in the **Font Directory** box is that of the system fonts. Otherwise, it is the location of the last font that was converted.

After a valid directory is selected, the available font names appear in the list box.

3 If necessary, specify a location to save the converted fonts in the **Output Directory field.**

Converted fonts are saved in <product_root>\Resources\lib\fsfiles. You don't have to change the output directory because it should be set correctly by default.

4 In **Fonts, select the font to convert.**

The **Preview** displays what a selected font looks like.

5 In **Conversion Options, select the font styles (i.e., **Plain**, **Bold**, **Italic**, and **Bold Italic**) to convert.**

Many TrueType fonts (but not all) contain information about several different font styles. Most of the time, these styles are *plain*, *italic*, *bold*, and *bold italic*.

Since each Corda font style is stored in a separate file, each style can be converted separately. By default, the Corda Font Converter assumes you want to convert all of the styles. If this is not the case, uncheck any styles that should not be converted.

6 In **Display Name, specify a name for the converted font by typing it into the box.**

- ADDITIONAL INFORMATION

- Corda Fonts

By default, the converted font has the same name as the TrueType font, but this name can be whatever you want it to be. Corda Builder and Corda Server only know the font by the name assigned here.

7 Click **Convert to create the new Corda font.**

Corda Font Converter converts the font. The **Status** box indicates the progress of the conversion process. When the box reports *Conversion Process Completed successfully*, the conversion process is done.

If the **Status** box reads *Conversion Process Failed*, Corda Font Converter is unable to convert the specified font; try a different font.

The converted font files are saved in <product_root>\Resources\lib\fsfiles, unless you have specified an alternate location. The file names have eight characters—some combination of the font style and name.

8 To convert another font, repeat steps 2 - 7.

When finished, close the Corda Font Converter by clicking **Close**.

9 To use the new font in a Corda application, restart that application.

Note: *The Corda Font Converter is based on the TrueType-to-SVG font converter developed by Steady State Software Limited (<http://www.steadystate.com/>). The source code for the font converter is available and released under the GNU Lesser General Public License (see [docs/other/gnulicense.html](#)). To obtain source code, contact Corda Technologies at the support number listed at the front of this Guide.*

INSTALLING CUSTOM FONTS

If you created the custom font, it is already installed for use by Corda Builder. Otherwise, follow these steps to install a custom font for Corda Builder.

Important: *These instructions pertain to installing a custom font in Corda Builder. Although the directions for installing to **Corda Server** are similar, see [“Installing Custom Fonts” on page 9-6](#) of the *Corda Installation and Administration Guide* for instructions on installing fonts for **Corda Server**.*

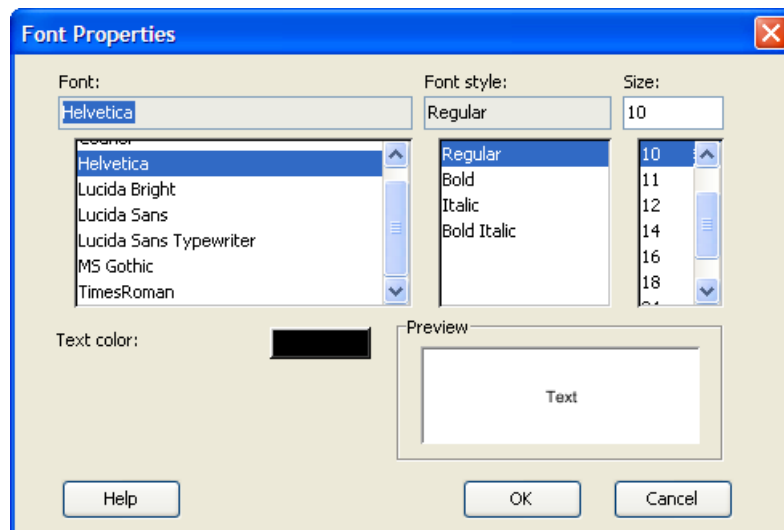
To install pre-converted fonts

- 1 Locate the custom fonts.**
Corda fonts files have an .fsd extension.
- 2 Copy the custom font files to the <product_root>\Resources\lib\fsfiles directory.**
- 3 Restart Corda Builder so that it recognizes the new font files.**
Upon restart, the fonts are available for use.

USING CUSTOM FONTS

Important: *If you reference a custom font in an Image Template file for use with Corda Server, upload the custom font to the appropriate Corda Server. Otherwise, Corda Server is unable to publish Image Template files with custom fonts. For more information, see [“Installing Custom Fonts” on page 9-6](#) of the Corda Server Installation and Administration Guide.*

After you install a custom font (see [“Installing Custom Fonts” on page 11-34](#)), it is available to any Corda Builder projects. Use the **Font** property to select a font and configure it for specific needs. Select the ellipsis [...] button to open the **Font Properties** dialog, which allows you to select fonts and font characteristics from dropdown lists.



Specify font characteristics directly in **Object Properties**.

Properties and attributes related to custom fonts are available from the **Font** property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

ADDITIONAL INFORMATION

Corda Fonts

INTERNATIONAL FONTS AND CHARACTER SUPPORT

This section discusses the use of Corda Builder with international (non-U.S.) fonts and character sets. In this section you find the following topics:

- [Installing International Fonts](#)
- [Using Non-Latin Characters in Projects](#)
- [Changing the Corda Builder Dialog Font](#)

INSTALLING INTERNATIONAL FONTS

The Times, Helvetica, and Courier font sets included with Corda Builder provide very little support for international characters by default. The Corda Builder Lucida font provides some international support but does not contain any Asian or double-byte characters.

Because of this, you might need to install international fonts to Corda Builder before taking full advantage of Corda Builder's international and double-byte character support.

If necessary, create international fonts as you would any other custom font (see [“Creating Custom Fonts” on page 11-32](#)) and then install them on the system from which you are running Corda Builder (see [“Installing Custom Fonts” on page 11-34](#)).

Note: *A common font that contains a large number of Unicode characters is Arial Unicode MS.*

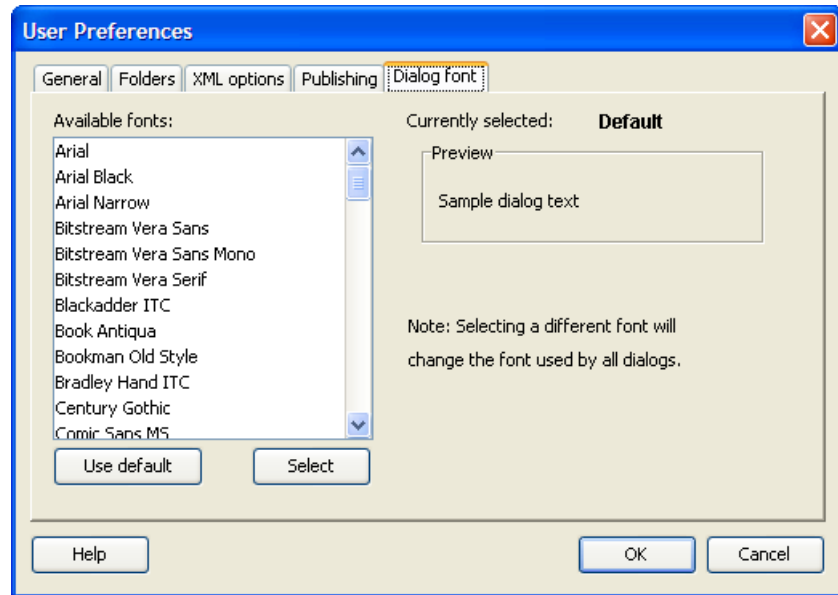
USING NON-LATIN CHARACTERS IN PROJECTS

After you have installed an international font, select it in Corda Builder as you would any other font. As long as the Image Template text uses an international font, use international (non-Latin) characters in it without any problems.

CHANGING THE CORDA BUILDER DIALOG FONT

If a box is displayed in place of a given non-Latin character in the Corda Builder interface, the Corda Builder dialog font is incapable of displaying the specified international character. However, if the image displays the correct character in the preview, try changing

the Corda Builder dialog font, in **Edit > Preferences > Dialog Font**, to something that displays the character properly.



- **ADDITIONAL INFORMATION**
- *Corda Fonts*
-
-



IMAGE FORMATS

Corda® Server™ can generate Corda images in the following formats: Macromedia Flash*, SVG*, PNG, JPEG, PDF, EPS, TIFF, and EMF. This chapter describes these formats in detail and tells you how to generate a Corda image in each particular format. The last section of this chapter, “[Comparison of Image Format Features](#),” has a convenient table comparing all of the features of the different formats.

- IMAGE FORMATS
- Macromedia Flash
-
-

MACROMEDIA FLASH

Flash is a vector graphic image type developed by Macromedia and is widely used in Web sites. Flash images are small, load quickly, and support interactive capabilities such as drilldown, rollover, and hover text. Flash images also support animations and auto-updating. Flash images print at a high resolution. Another advantage is that a user can put a Flash image directly into a Microsoft PowerPoint* presentation.

The Flash viewer (Flash 3.0 or later) is required to view a Flash image. It is estimated that more than 96% of Internet users already have the Flash viewer.

More information about the Flash format is available from Macromedia's Web site at <http://www.macromedia.com/software/flash/>.

Note: Best Image Fallback lets a user's Web browser dynamically decide what image format to display. In this case, it probably does not matter if you choose Flash or SVG as the format of the Corda image. The user sees the Corda image in JPEG format if he or she doesn't have the Flash or SVG plug-in. For more information, see "Best Image Fallback" on page 8-3 of the Corda 7 Developer Reference.

EMBEDDING A FLASH IMAGE USING CORDA EMBEDDER

Generate a Flash Corda image by setting the Corda Embedder `outputType` attribute to "FLASH." For example, assuming the Corda Embedder object is named `myImage`, you use the following command to set the image format to Flash:

```
myImage.outputType = "FLASH";
```

EMBEDDING A FLASH IMAGE IN HTML

Generate a Flash Corda image using the `@_FLASH` server command. For example, the following HTTP request generates a Flash Corda image:

```
http://<server
address>:2001/?@_FILEexamples\bar.itxml@_FLASH
```

Flash images can be embedded in Web pages using an `<object>` tag for Microsoft Internet Explorer browsers and an `<embed>` tag for Netscape* browsers. Use the `@_FLASH` server command to instruct Corda Server to generate a Flash image.

The following code demonstrates how to embed a Flash image. Note that by embedding the `<embed>` tag within the `<object>` tag, the code makes sure that the image displays correctly on both Microsoft Internet Explorer and Netscape.

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
  codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=4,0,0,0"
  width="600" height="400">
  <param name="MOVIE" value="http://<server
address>:2001/?@_FILEexamples\bar.itxml@_HEIGHT400@_
WIDTH600@_FLASH">
  <embed type="application/x-shockwave-flash"
  pluginspage="http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_Version=ShockwaveFlash"
  width="600" height="400" src="http://<server
address>:2001/?@_FILEexamples\bar.itxml@_HEIGHT400@_
WIDTH600@_FLASH">
</embed></object>
```

Implement *Best Image Fallback* (i.e., an alternate image if the browser is incapable of downloading the Flash plug-in) to a JPEG or PNG image by embedding an `` tag within the `<embed>` tag. For more information on embedding an image in the `` tag, see [“Embedding a Corda Image in an Image Tag” on page E-5 of the Corda 7 Developer Reference](#).

Note: *It is much more convenient to embed a FLASH image with the Corda Embedder.*

AUTO-UPDATING FLASH IMAGES

Corda Server supports auto-updating Flash images. This feature lets you do two different things: display a single Corda image that refreshes itself automatically, or display a series of Corda images in a repeating loop. By using this feature, you can automatically display related and up-to-date images without the need for any client-side scripting.

Warning: *The automatically updated images feature is not compatible with animation.*

Auto-updating supports only Flash images. Use the following properties to configure and enable auto-updating:

URL LIST: Specifies one or more URLs, or file names, that the Image Template file displays. Specifying a single URL results in continually updated Flash images or content. Specifying multiple URLs creates a repeating loop of different Flash images or content.

- **IMAGE FORMATS**

- *Macromedia Flash*
-
-

PLAYBACK TIME: Specifies an amount of time, in seconds, that the Image Template file displays each URL in the URL List before re-loading an updated URL or loading the next URL in the URL List.

PLAYBACK SPEED: When a URL consists of multiple frames, like a Flash movie, Playback speed specifies the number of frames to display per second. A typical setting for smooth playback is 20 (frames per second).

More information on these properties is available in the [Corda 7 Object Reference](#).

When a browser loads the initial Flash image (the one based on the Image Template file in which you specified the auto-updating settings), the image instructs the Flash plug-in to cycle through the URLs named in the Image Template file. These URLs can point to any Flash image you want to point to—even Flash images not generated by Corda Server.

The Flash plug-in cycles through these images continuously, using the auto-update interval specified in the Image Template file. This means, for example, if you only specify one Flash image URL, the plug-in continuously refreshes the image from the URL at the specified interval.

Note: *Don't specify auto-update settings in any of the Flash images you are loading. These only need to be set in the initial project file that is being loaded into the Web page.*

Consider the following example: In an Image Template file that you are using to generate a Corda image in a Web page, you specify the following URLs in the auto-update list: `http://myserver.com/getgraph?index=1`, `http://myserver.com/getgraph?index=2`, and `http://myserver.com/getgraph?index=3`. Suppose the time interval for updating is 20 seconds.

When you first load the graph- or map-enabled Web page, the browser loads the base Corda image (from the Image Template file) into the Flash plug-in. After 20 seconds, it loads a Flash image from `http://myserver.com/getgraph?index=1`. After another 20 seconds, it loads a Flash image from `http://myserver.com/getgraph?index=2`. After another 20 seconds, it loads a Flash image from `http://myserver.com/getgraph?index=3`. The plug-in then continues to cycle through these three URLs until the browser leaves the page.

Important: *All of the URLs in the auto-update list must return Flash (.swf) files. Otherwise, the plug-in is not able to display the URL.*

Note that if a Flash image stored at one of these URLs changes, you see the changes in the auto-updating Flash image. So, for instance, if the data used to create the graph in `http://myserver.com/getgraph?index=2` changes, the graph shown in the auto-updating Flash image changes accordingly, giving the graph the appearance of streaming its data.

Important: *On some browsers, you may experience caching problems with auto-updating Flash images. To get around these problems, set the `Content Expiration` header for each Flash image you load to expire immediately. See [“Solving Caching Issues with Auto-Updating Flash images”](#) on page 12-5 for details.*

SOLVING CACHING ISSUES WITH AUTO-UPDATING FLASH IMAGES

If you dynamically update Flash images, but do not see the changes reflected in the auto-updating Flash image, you might be experiencing Web browser caching problems.

When a Web server sends a file (whether it be an image or a Web page) to a browser, it includes some instructions—in other words, a header—that help the browser know how to deal with that file. One piece of information it sends is the content-expiration header—that is, how long the browser should wait before checking the server to see if the file has been updated. If you request an image or Web page again, but the content is not expired, the browser simply shows you the file that it received the last time it requested the image. Otherwise, it contacts the server to see if there is an updated file to download.

Some browsers make assumptions about content-expiration that can cause the browser not to check for updated content in auto-update Flash images. There are a couple of ways to fix this problem:

- [Changing Browser Cache Settings](#)
- [Setting the Content Expiration for Flash Images](#)

CHANGING BROWSER CACHE SETTINGS

An easy (though understandably less desirable) solution to this problem is to change the browser's settings so that it always checks for newer versions of Web pages and images.

To do this on Internet Explorer, for example, do the following:

- 1 **Select *Tools* > *Internet Explorer*.**
- 2 **Under *Temporary Internet Files*, click the *Settings* button.**
- 3 **Under the *Check for Newer Versions of Stored Pages* option, select *Every Visit to the Page*.**

The main drawback of this solution is that it requires you to instruct everyone who views the image to complete the same procedure.

SETTING THE CONTENT EXPIRATION FOR FLASH IMAGES

To avoid having to rely on the clients to change their browser settings, modify the content-expiration header for each Flash image you load so that it expires immediately. By setting Flash images to expire immediately, you force the browser to always check for updated images.

IMAGES STORED ON A WEB SERVER When loading Flash images that have been saved to a Web server, set the content-expiration for the image on a file-by-file basis. The exact procedure for setting the content-expiration depends on the Web server. Some servers might not provide a mechanism for setting content expiration. For example, Windows IIS uses the following procedure:

- IMAGE FORMATS
- Macromedia Flash
-
-

- 1 Open the **Internet Information Services** management console.
- 2 Using the tree on the left, browse to the Flash image file whose content-expiration settings you want to change.
- 3 Right-click the Flash image file and select **Properties**.
- 4 Select the **HTTP-Headers** tab.
- 5 Check the **Enable Content Expiration** box.
- 6 Under **Content Should**, select **Expire Immediately**.

IMAGES SERVED FROM CORDA SERVER When retrieving Flash images from Corda Server using server commands, use the `@_DONTCACHE` command to force the content to expire immediately, as illustrated below:

```
http://<server
address>:2001/?@_FILEexamples\bar.itxml@_DONTCACHE
```

When loading a previously saved image (see [“Loading Saved Files with Server Commands” on page 12-7](#)), use the following command:

```
http://localhost:2001/?@_LOADimages/test.swf@_Flash@_D
ONTCACHE
```

IMAGES RETURNED VIA A WEB APPLICATION When using a Web application to output a Flash image directly to the browser (for example, using the **Corda Embedder** `getBytes()` command), programmatically set the content-expiration header.

The following line of code shows how to do this in Java:

```
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Pragma", "no-cache");
response.setDateHeader("Expires", 0);
```

The following line of code shows how to do this in ASP.NET:

```
Response.Cache.SetExpires(DateTime.Now);
```

See the Web application documentation for more information on how to do this in the Web environment.

SVG

Only available with PopChart Enterprise.

Scalable Vector Graphics (SVG) is an XML-based image format created and adopted by the World Wide Web Consortium (W3C) as the standard for vector graphic images. SVG images can be zoomed in and out without losing any details and have smaller file size than do JPEG, Flash, or PNG images. SVG makes possible high-resolution printing, animation, drilldown, rollover, and hover text along with other special effects. SVG is an open standard.

More information about the SVG format is available from Adobe's Web site at <http://www.adobe.com/svg/>.

Note: *With Best Image Fallback, have the user's Web browser dynamically decide what image format to display. In this case, it probably does not matter if you choose Flash or SVG as the format of the Corda image. The user sees the Corda image in JPEG format if he or she doesn't have the Flash or SVG plug-in. For more information, see "Best Image Fallback" on page 8-3 of the Corda 7 Developer Reference.*

EMBEDDING AN SVG IMAGE USING CORDA EMBEDDER

Generate an SVG Corda image by setting the Corda Embedder `outputType` attribute to "SVG." For example, assuming the Corda Embedder object is named `myImage`, you use the following command to set the image format to SVG:

```
myImage.outputType = "SVG";
```

When trying to generate an SVG image in a format that can be processed by Apache* FOP (Formatted Object Processor), set the `outputType` attribute to "SVGFOF" instead.

EMBEDDING AN SVG IMAGE IN HTML

Generate an SVG Corda image using the `@_SVG` server command. For example, the following HTTP request generates an SVG Corda image:

```
http://<server  
address>:2001/?@_FILEexamples\bar.itxml@_SVG
```

- IMAGE FORMATS
- SVG
-
-

SVG images can be embedded in Web pages using an `<embed>` tag and the `@_SVG` server command. You must also specify the width and the height of the image in the `<embed>` tag:

```
<embed type="image/svg+xml"
  pluginpage="http://www.adobe.com/svg/viewer/install
  /main.html" width="600" height="400"
  src="http://<server
  address>:2001/?@_FILEexamples\bar.itxml@_HEIGHT400@_
  WIDTH600@_SVG">
</embed>
```

When trying to generate an SVG image in a format that can be processed by Apache FOP (Formatted Object Processor), use the `@_SVGFOF` command instead.

PNG

The Portable Network Graphics (PNG) format is a lossless image format. Its file size is slightly larger than FLASH and JPEG and significantly larger than SVG. However, the quality is significantly less than that of these formats. PNG images print only at a low resolution of 72 DPI.

Corda Server uses Javascript* to provide interactivity such as drilldown and hover text in PNG images. In many cases, interactivity is just as good as it is in a Flash or SVG image. If you prefer not to use Javascript to provide interactivity, Corda Server can provide limited drilldown and hover capabilities via an image map. For more information, see [“Drilldown / Hover / Roll-Over Data Label Problems with PNG or JPEG Images” on page 11-11](#) of the *Corda 7 Install and Administration Guide*. PNG images cannot show hover objects or animations.

The PNG format is supported by most Web browsers, including Netscape browsers version 4.0 or later, and Microsoft browser’s version 4.0.1 or later, and does not require a plug-in. PNG images are anti-aliased.

AUTOMATIC PNG DETECTION

One significant feature of Corda Server is Automatic PNG Detection. The Corda Server can detect from a client’s request whether or not a Web browser supports PNG images. When this feature is enabled and a client requests an image in the “AUTO” format, Corda Server returns a PNG image if the browser supports PNG; otherwise, it returns a JPEG image.

Automatic PNG Detection (see [“Automatic PNG Detection” on page 3-26](#) of the *Corda 7 Install and Administration Guide*) must be enabled in the Administration Console for Corda Server to automatically return the most appropriate image format.

EMBEDDING A PNG IMAGE USING CORDA EMBEDDER

If Automatic PNG Detection is enabled, you usually embed a PNG image by requesting an “AUTO” image instead. However, to explicitly request a PNG Corda image, set the Corda Embedder `outputType` attribute to "PNG." For example, assuming the Corda Embedder object is named *myImage*, use the following command to set the image format to PNG:

```
myImage.outputType = "PNG";
```

- IMAGE FORMATS
- PNG
-
-

EMBEDDING A PNG IMAGE IN A WEB PAGE

If Automatic PNG Detection is enabled, embed a PNG image by requesting an “AUTO” image instead, or explicitly generate a PNG Corda image using the @_PNG server command. For example, the following HTTP request generates a PNG Corda image:

```
http://<server
address>:2001/?@_FILEexamples\bar.itxml@_PNG
```

If you have enabled Automatic PNG Detection (see [“Automatic PNG Detection” on page 3-26](#) of the *Corda 7 Install and Administration Guide*), the following request also creates a PNG image, as long as the browser supports PNG:

```
http://<server
address>:2001/?@_FILEexamples\bar.itxml@_AUTO
```

Embed a PNG image in a Web page simply by making it the source of an image tag. For example, the following image tag embeds the PNG image we created with the HTTP request above. It also provides an alternate text description of *My Corda Image* in case the image loads slowly or the user is visually impaired.

```

```

JPEG

JPEG (or JPG) stands for Joint Photographic Experts Group and was named for the organization that developed the format. It is an image format that is almost universally supported. It allows for millions of colors; however, because it is a lossy format, JPEG images may look fuzzier and less crisp than images in other formats. JPEG images are larger in file size than SVG or Flash images. They are typically smaller than PNG image files but also of lesser quality. JPEG images generated by Corda Server print only at a low resolution of 72 DPI.

Corda Server uses Javascript to provide interactivity such as drilldown and hover text in JPEG images. In many cases, interactivity is just as good as it is in a Flash or SVG image. If you prefer not to use Javascript to provide interactivity, Corda Server can provide limited drilldown and hover capabilities via an image map. For more details, see [“Drilldown / Hover / Roll-Over Data Label Problems with PNG or JPEG Images”](#) on page 11-11 of the *Corda 7 Install and Administration Guide*. JPEG images cannot show hover objects or animations.

By default, JPEG images are compressed at a level of 7.5 on a scale of 0-10, where 0 is the highest compression and 10 is no compression. Change the Corda Server compression level by modifying the `<CS_JPGQuality>` server configuration setting. JPEG images are anti-aliased.

EMBEDDING A JPEG IMAGE USING CORDA EMBEDDER

Generate a JPEG Corda image by setting the Corda Embedder `outputType` attribute to "JPEG." For example, assuming the Corda Embedder object is named *myImage*, you use the following command to set the image format to JPEG:

```
myImage.outputType = "JPEG";
```

EMBEDDING A JPEG IMAGE IN A WEB PAGE

Generate a JPEG Corda image using the `@_JPEG` server command. For example, the following HTTP request generates a JPEG Corda image:

```
http://<server  
address>:2001/?@_FILEexamples\bar.itxml@_JPEG
```

- IMAGE FORMATS
- JPEG
-
-

Embed a JPEG image in a Web page simply by making it the source of an image tag. For example, the following image tag embeds the JPEG image we created with the HTTP request above. It also provides an alternate text description of *My Corda Image* in case the image loads slowly or the user is visually impaired.

```

```


PDF

Available only with Corda 7 Enterprise.

As its name implies, Portable Document Format (PDF) is more of a document format than an image format. Because of its high quality, ease of printing, relatively small file size (its size is comparable to Flash), and widespread use, PDF is an attractive format for displaying Corda images.

Corda images in the PDF format are somewhat limited. They offer no drilldown or hover capabilities and can't display transparency. Another drawback is that PDF images are difficult to embed in Web pages. Some browsers display the image in an embedded Acrobat Reader* interface that may be distracting or confusing to some users. Other browsers can't embed PDF documents at all. Most of the time, you want to link to the PDF image rather than embed it in a Web page.

PDF documents may be viewed in a wide variety of viewers, including the free Acrobat Reader, included on the installation CD or available from Adobe at <http://www.adobe.com>. PDF documents may be viewed in Web browsers by using a plug-in (such as the one that is automatically included with Acrobat Reader). Such a plug-in is already installed on the large majority of Web browsers.

EMBEDDING A PDF DOCUMENT USING CORDA EMBEDDER

Generate a PDF Corda image by setting the Corda Embedder `outputType` attribute to "PDF." For example, assuming the Corda Embedder object is named `myImage`, you use the following command to set the image format to PDF:

```
myImage.outputType = "PDF";
```

Because of their limitations, most people don't embed PDF documents in Web pages. Instead, they create PDF Corda images that can be served by themselves or can be embedded in other PDF documents. If this is the intent, you probably are interested in the `getBytes()` method, which allows you to access a byte array of the PDF image and even return a PDF image directly from a Java Servlet.

Another alternative is to save a PDF image for future viewing with either the `saveToAppServer` or the `saveToCordaServer` Corda Embedder methods.

Use the Corda Server Java Library to create PDF images for non-Web applications. To learn how to create a Corda image using the Corda Server Java Library, contact the Corda Technologies support team (support@corda.com).

- IMAGE FORMATS
- PDF
-
-

EMBEDDING A PDF DOCUMENT IN HTML

Request that Corda Server generate an image in the PDF format by using the `@_PDF` server command. For example, assuming the Corda Server is running on the local system with its default settings, the following HTTP request generates a PDF Corda image:

```
http://<server
address>:2001/?@_FILEexamples\bar.itxml@_PDF
```

If you attempt to browse to a PDF document—for example, entering the location above in the location bar of a browser—a plug-in, such as Acrobat Reader, most likely starts up. Depending on the browser and the version of the plug-in, the plug-in may start as a separate program or it may start within the browser window.

If you do not have the proper plug-in, you are asked to specify a location to download the file. In some instances, despite the fact that you don't have the plug-in, you may still have software capable of opening the downloaded document.

Most people don't embed PDF documents within a Web page. Instead, they provide a link to the PDF document, which can be viewed separately. [Example 12.1](#) shows how you might link to a PDF Corda image.

Example 12.1 Linking to a PDF Document

```
<a href="http://<server
address>:2001/?@_FILEexamples\bar.itxml@_PDF">
Click here to view the graph in PDF format
</a>
```

When a viewer clicks on a PDF link, the browser either jumps to the PDF document or, if the proper plug-in is not installed, downloads the document.

If you want to embed the PDF Corda image within a Web page, use one of the following techniques:

EMBED THE PDF USING THE `<object>` TAG: Specify the PDF Corda image as the source of this object (using the `src` attribute). This technique works in most current Web browsers, including Microsoft Internet Explorer 5.0 and up, and Netscape Navigator 4.08 and up. However, the results may be different based upon the specific Web browser used. For example, with Internet Explorer, the Acrobat Reader interface is used to display the PDF document, so the Acrobat Reader menu and control bars are displayed as part of the “image.” Netscape browsers, on the other hand, display the entire PDF document as an image, without PDF menus or control bars; however, the user is unable to resize or save the image.

Example 12.2 illustrates this technique.

Example 12.2 Using an `<embed>` Tag to Embed a PDF Document

```
<embed height="300" width="400" src="http://<server
address>:2001/?@_FILEexamples\bar.itxml@_PDF">
</embed>
```

EMBED THE PDF DOCUMENT USING AN `<IFRAME>` TAG: If a user's browser is current, including Microsoft Internet Explorer 5.0 or above, or Netscape 6.1 or above, the `<iframe>` tag creates a frame within the current viewing window. Specify the PDF Corda image as the frame source (using the `src` attribute). When using the Acrobat Reader plug-in, this technique results in the Acrobat Reader menu and control bars being shown within the frame, regardless of the Web browser being used.

Example 12.2 illustrates this technique. It also provides a link to the PDF document in case the user's browser does not support the `<iframe>` tag.

Example 12.3 Using an `<iframe>` Tag to Embed a PDF Document

```
<p>The graph in the PDF document below plots last month's
sales.</p>
<iframe height="300" width="400" src="http://<server
address>:2001/?@_FILEexamples\bar.itxml@_PDF">
[If you see this text, the browser cannot display this graph inside
of this document. Click <a href="http://<server
address>:2001/?@_FILEexamples\bar.itxml@_PDF">here</
a> to view it separately.]
</iframe>
```

- IMAGE FORMATS
- PDF
-
-

FONTS IN PDF

The PDF format supports embedding font definitions directly in a PDF file so, even if a system does not have access to the fonts used in a PDF document, it still displays correctly because the PDF font definitions are always available.

Corda Server embeds most fonts within its PDF output:

- [Automatically Embedded Fonts](#)
- [Non-Embedded \(Standard\) Fonts](#)

AUTOMATICALLY EMBEDDED FONTS

Corda Server automatically embeds most fonts within its PDF output. It retrieves font definitions from TrueType* fonts installed on the system on which it is running. Corda Server looks for fonts in the directories specified in its `pdf_config.txt` file.

If a requested font cannot be found on the system on which Corda Server is running, Corda Server uses the default font (TimesRoman) instead. It also uses the default font if a requested font's license does not permit the embedding of the font. A warning message appears in the Corda Server console output under either circumstance.

See [“To Add or Modify the Embedded Fonts Directory”](#) on page 9-11 of the *Corda 7 Install and Administration Guide* to learn how to change or add font directories in the server configuration file.

NON-EMBEDDED (STANDARD) FONTS

Standard fonts, by default, are not embedded in Corda Server PDF output. These fonts include Courier, Helvetica, and TimesRoman. These fonts are included automatically in most PDF viewing software, including Adobe Reader, and thus in most cases do not need to be embedded.

When Corda Server gets a request for a font with the name *helvetica*, *timesroman*, or *courier*, it assumes that the font is one of these standard fonts and it does not embed it. This means that any fonts in PopChart or OptiMap Image Template files whose names consist of the words *Helvetica*, *Lucida Sans*, *TimesRoman*, *Lucida Bright*, *Courier*, or *Lucida Sans Typewriter* is rendered in PDF using standard fonts that are not embedded within the outputted file. This also means that fonts in HTML/XHTML documents with names consisting of *helvetica*, *sans-serif*, *fantasy*, *timesroman*, *times*, *serif*, *cursive*, *courier*, or *monospaced* are rendered in PDF with standard fonts that are not embedded within the outputted file.

Generally, you do not need to worry about embedding standard fonts. However, in certain cases you may want to force Corda Server to embed standard fonts. See [“To Force Standard Fonts To Be Embedded”](#) on page 9-13 of the *Corda 7 Install and Administration Guide* to learn how to do this.

EPS

Available only with PopChart Enterprise.

Encapsulated PostScript (EPS) is a standard file format for importing and exporting PostScript files. A Corda image in this format is essentially a single page PostScript file that describes the graph or map.

The EPS format is very similar to PDF. Consequently, Corda images in EPS format are the same size as images in PDF. They are of very high quality, but like PDF, have no drilldown or hover capability and can't display transparency.

The main purpose of an EPS file is to be included in other applications or documents. EPS images cannot be viewed from a Web browser and must instead be viewed from a graphics manipulation program, such as Adobe PhotoShop*. They cannot be embedded in a Web page.

The Corda Server EPS capabilities are not applicable for most users. Unless you are aware of a reason to produce Corda images in the EPS format, you probably do not need to worry about this format.

EMBEDDING AN EPS IMAGE USING CORDA EMBEDDER

EPS images cannot be embedded in a Web page—they can only be downloaded and viewed with an external viewer. One common practice is to create a scheduled task to generate an EPS image at regular intervals and then provide a link to it for users to download.

Generate an EPS Corda image by setting the Corda Embedder `outputType` attribute to "EPS." For example, assuming the Corda Embedder object is named `myImage`, you use the following command to set the image format to EPS:

```
myImage.outputType = "EPS";
```

This creates an EPS Corda image that can be downloaded by itself or can be embedded in other EPS documents. If this is the intent, you probably are interested in the `getBytes()` method, which allows you to access a byte array of the EPS image and even return an EPS image directly from a Java Servlet.

Another alternative is to save an EPS image for future viewing using either the `saveToAppServer` or the `saveToCordaServer` Corda Embedder methods. For example, you can create a scheduled task to generate an EPS image at regular intervals and then provide a link to it for users to download. If you save the EPS image to the Corda Server, load the image in a different session with the `loadFromCordaServer` method.

- IMAGE FORMATS
- EPS
-
-

Use the Corda Server Java Library to create EPS images for non-Web applications. To learn how to create a Corda image using Corda Server Java Library, contact the Corda Technologies support team (support@corda.com).

EMBEDDING AN EPS IMAGE IN HTML

Request that Corda Server generate an image in the EPS format by using the `@_EPS` server command. For example, the following HTTP request generates an EPS Corda image:

```
http://<server
address>:2001/?@_FILEexamples\bar.itxml@_EPS
```

If you attempt to browse to an EPS image—for example, entering the location above in the location bar of a browser—you are prompted to enter a location on the computer to download the file because you cannot view it in a browser. Save it, with an `.eps` extension, and then open it up in a suitable graphics program.

EPS images cannot be embedded in a Web page. Instead, provide a link to the EPS image that can be viewed separately. [Example 12.4](#) shows how you might link to an EPS image.

Example 12.4 Linking to an EPS Image

```
<a href="http://<server
address>:2001/?@_FILEexamples\bar.itxml@_EPS">
Click here to view the graph in EPS format
</a>
```

When a viewer clicks on the EPS link, the Web browser downloads an EPS file containing the Corda image.

Accessing an EPS Corda image in any way other than as a file download requires a proprietary system. To do this, you probably want to generate the EPS Corda images using the methods outlined in the previous section.

FONTS IN EPS

Corda Server does not embed fonts in its EPS output. Text in the following fonts are referenced in the Corda Server EPS output: *Times*, *Courier*, and *Helvetica*. It is assumed that these fonts are available in most situations.

Text in all other fonts, including fonts that you import with the Corda Font Converter, are drawn manually. In other words, each character is drawn as a vector—from a technical standpoint they aren't characters at all; they're images.

Fonts that Corda Server has to draw may look bad on screen but print well. Fonts that Corda Server references (Times, Courier, and Helvetica), however, look great on screen and in print.

USING FONT DRAWING INSTEAD OF REFERENCING

In some publishing processes, the fonts that Corda Server references (Times, Courier, and Helvetica) may be unavailable. If this is the case, you can have Corda Server draw these fonts as well. If you use this setting, these fonts look bad on screen but won't cause problems during publishing processes.

To use this setting, set the image type to *EPS_CF*, as illustrated below:

```
myImage.outputType = "EPS_CF";
```

Or use the *@_EPS_CF* server command, as illustrated below:

```
http://<server  
address>:2001/?@_FILEexamples\bar.itxml@_EPS_CF
```

- IMAGE FORMATS
- TIFF
-
-
-

TIFF

Available only with PopChart Enterprise.

Tagged Image File Format (TIFF) is a standard file format designed to promote the interchange of digital image data. TIFF is a 24-bit, raster data, lossless format, used chiefly to store the “master” copy of images used in desktop or photo publishing.

TIFF Corda images are of high quality but, like PDF, have no drilldown or hover capability. Corda Server does not compress TIFF images, and consequently they are typically between 300 to 700 KB in size. However, they usually compress well using LZW or ZIP compression methods (available through save options within most graphics manipulation programs).

TIFF images cannot be viewed from a Web browser and must instead be viewed from a graphics manipulation program, such as Adobe PhotoShop. They cannot be embedded in a Web page.

The Corda Server TIFF capabilities are not applicable for most users. Unless you are aware of a reason to produce Corda images in the TIFF format, you probably do not need to worry about this format.

EMBEDDING A TIFF IMAGE USING CORDA EMBEDDER

TIFF images cannot be embedded in a Web page—they can only be downloaded and viewed with an external viewer. One common practice is to create a scheduled task to generate a TIFF image at regular intervals and then provide a link to it for users to download.

Generate a TIFF Corda image by setting the Corda Embedder `outputType` attribute to "TIFF." For example, assuming the Corda Embedder object is named `myImage`, you use the following command to set the image format to TIFF:

```
myImage.outputType = "TIFF";
```

This creates a TIFF Corda image that can be downloaded by itself or can be embedded in documents. If this is the intent, you probably are interested in the `getBytes()` method, which allows you to access a byte array of the TIFF image and even return a TIFF image directly from a Java Servlet.

Another alternative is to save a TIFF image for future viewing using either the `saveToAppServer` or the `saveToCordaServer` Corda Embedder methods. For example, you can create a scheduled task to generate a TIFF image at regular intervals and then provide a link to it for users to download. If you save the TIFF image to the Corda Server, load the image in a different session with the `loadFromCordaServer` method.

Use the Corda Server Java Library to create TIFF images for non-Web applications. To learn how to create a Corda image using the Corda Server Java Library, contact the Corda Technologies support team (support@corda.com).

EMBEDDING A TIFF IMAGE IN HTML

Request that Corda Server generate a TIFF image by using the `@_TIFF` server command. For example, the following HTTP request generates a TIFF Corda image:

```
http://<server
address>:2001/?@_FILEexamples\bar.itxml@_TIFF
```

If you attempt to browse to a TIFF image—for example, entering the location above in the location bar of a browser—you are prompted to enter a location on the computer to download the file because you cannot view it in a Web browser. Save the file with a `.tif` extension and then open it in a suitable graphics program.

TIFF images cannot be embedded in a Web page. Instead, provide a link to the TIFF image so it can be viewed separately. [Example 12.5](#) shows how you might link to a TIFF image.

Example 12.5 Linking to a TIFF Image

```
<a href="http://<server
address>:2001/?@_FILEexamples\bar.itxml@_TIFF">
Click here to view the graph in TIFF format
</a>
```

When a viewer clicks on this link, the browser downloads a TIFF file containing the Corda image.

Accessing a TIFF Corda image in any way other than as a file download requires a proprietary system. To do this, you probably want to generate the TIFF Corda images using the methods outlined in the previous section.

- IMAGE FORMATS
- EMF
-
-

EMF

Available only with PopChart Enterprise.

Enhanced Meta File (EMF) is a vector-based image format designed for and popularized by Microsoft Windows. EMF Corda images are of high quality but, like PDF, have no drilldown or hover capability. They also do not support transparency. EMF files are typically between 5 to 10 KB in size.

EMF images cannot be viewed from most browsers and must instead be viewed from a graphics manipulation program, such as Adobe PhotoShop. They cannot be embedded in a Web page.

The Corda Server EMF capabilities are not applicable for most users. Unless you are aware of a reason to produce Corda images in the EMF format, you probably do not need to worry about this format.

EMBEDDING AN EMF IMAGE USING CORDA EMBEDDER

EMF images cannot be embedded in a Web page—they can only be downloaded and viewed with an external viewer. One common practice is to create a scheduled task to generate an EMF image at regular intervals and then provide a link to it for users to download.

Generate an EMF Corda image by setting the Corda Embedder `outputType` attribute to "EMF." For example, assuming the Corda Embedder object is named `myImage`, you use the following command to set the image format to EMF:

```
myImage.outputType = "EMF";
```

This creates an EMF Corda image that can be downloaded by itself or can be embedded in documents. If this is the intent, you probably are interested in the `getBytes()` method, which allows you to access a byte array of the EMF image and even return an EMF image directly from a Java Servlet.

Another alternative is to save an EMF image for future viewing using either the `saveToAppServer` or the `saveToCordaServer` Corda Embedder methods. For example: you can create a scheduled task to generate an EMF image at regular intervals and then provide a link to it for users to download. If you save the EMF image to the Corda Server, load the image in a different session with the `loadFromCordaServer` method.

Use the Corda Server Java Library to create EMF images for non-Web applications. To learn how to create a Corda image using the Corda Server Java Library, contact the Corda Technologies support team (support@corda.com).

EMBEDDING AN EMF IMAGE IN HTML

Request that Corda Server generate an image in the EMF format by using the `@_EMF` server command. For example, the following HTTP request generates an EMF Corda image:

```
http://<server
address>:2001/?@_FILEexamples\bar.itxml@_EMF
```

If you attempt to browse to an EMF image—for example, entering the location above in the location bar of a Web browser—you are prompted to enter a location on the computer to download the file since you cannot view it in a Web browser. Save the file with an `.emf` extension and then open it in a suitable graphics program.

EMF images cannot be embedded in a Web page. Instead, provide a link to the EMF image so it can be viewed separately. [Example 12.6](#) shows how you might link to an EMF image.

Example 12.6 Linking to an EMF Image

```
<a href="http://<server
address>:2001/?@_FILEexamples\bar.itxml@_EMF">
Click here to view the graph in EMF format
</a>
```

When a viewer clicks on this link, the browser downloads an EMF file containing the Corda image.

Accessing an EMF Corda image in any way other than as a file download requires a proprietary system. To do this, you probably want to generate the EMF Corda images using the methods outlined in the previous section.

- **IMAGE FORMATS**

- *Comparison of Image Format Features*
-
-

COMPARISON OF IMAGE FORMAT FEATURES

This table provides a side-by-side comparison of image format features:

	meta tag media Flash	SVG	PNG	JPEG	PDF	EPS	TIFF
Typical Image Size	10 KB	4 KB	10-15 KB	15-25 KB	5-20 KB	5-20 KB	300-700 KB ^a
Number of Colors	Millions	Millions	Millions	Millions	Millions	Millions	Millions
High Resolution Printing	Yes	Yes	No (72 DPI)	No (72 DPI)	Yes	Yes	Yes
Interactivity (hover, drilldown, rollover)	Yes	Yes	Most browsers ^b	Most browsers ^c	No	No	No
Browser Support^d	96%	Limited	90%	100%	Most browsers	None	None

a. TIFF images can be compressed through external applications.

b. Interactivity in PNG images is supported via Javascript and might not be supported in older or non-standard browsers. Alternatively, interactivity can be supported by image maps, but only in a limited capacity. Interactivity with these image types is only available when using Corda Embedder.

c. Interactivity in JPEG images is supported via Javascript and might not be supported in older or non-standard browsers. Alternatively, interactivity can be supported by image maps, but only in a limited capacity. Interactivity with these image types is only available when using Corda Embedder.

d. With Best Image Fallback (see [“Best Image Fallback” on page 8-3](#) of the *Corda 7 Developer Reference*), browser support is irrelevant for Flash or SVG because the browser automatically displays the Corda image in the highest quality format available. Similarly, the Corda Server Automatic PNG Detection feature (see [“Automatic PNG Detection” on page 3-26](#) of the *Corda 7 Install and Administration Guide*) makes browser support irrelevant for PNG images.

TABLE OF MIME TYPES

The table below lists the mime types for the image formats that Corda Server supports.

Format	Mime Type
JPEG	image/jpeg
PNG	image/png
SVG	image/svg
TIFF	image/tiff
FLASH	application/x-shockwave-flash
PDF	application/pdf
EPS	application/postscript
EMF	image/emf

- **IMAGE FORMATS**
- *Table of MIME Types*
-
-



PUBLISHING THE PROJECT

This chapter discusses publishing a Corda Builder project as a Corda image or Web page.

This section guides you through the process of publishing a project as a static Corda image. The resulting image can be placed on a Web site, or used in other applications and documents. It also discusses how to use a project with Corda Server as an Image Template file for dynamically generated images.

Finally, this chapter discusses some issues related to published Corda images. Topics in this chapter include the following:

- [Publishing a Project](#)
- [Accessing Published Files](#)
- [Output Features](#)
- [Publishing Static Images on the Web](#)
- [Publishing Dynamic Image Template Files with Corda Builder's Sample Code](#)
- [Previewing a Corda Image](#)
- [Where Do I Get the SVG or FLASH Plug-In?](#)
- [How Do I Print a Corda Image?](#)
- [Tips for Creating Good Image Template Files](#)

- PUBLISHING THE PROJECT

- *Publishing a Project*

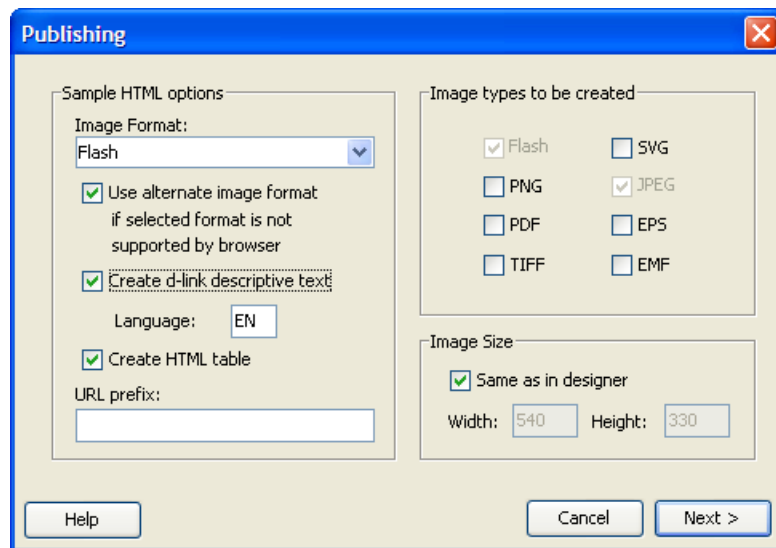
PUBLISHING A PROJECT

Although Corda Builder's most powerful use is for developing Image Template files that can be used to generate dynamic graph and map images on a Corda Server, static versions of Corda Builder projects can be published in up to eight different image formats.

To publish a static project

- 1 Load the desired Image template and select **File > Publish**.

This launches the **Publishing** dialog.

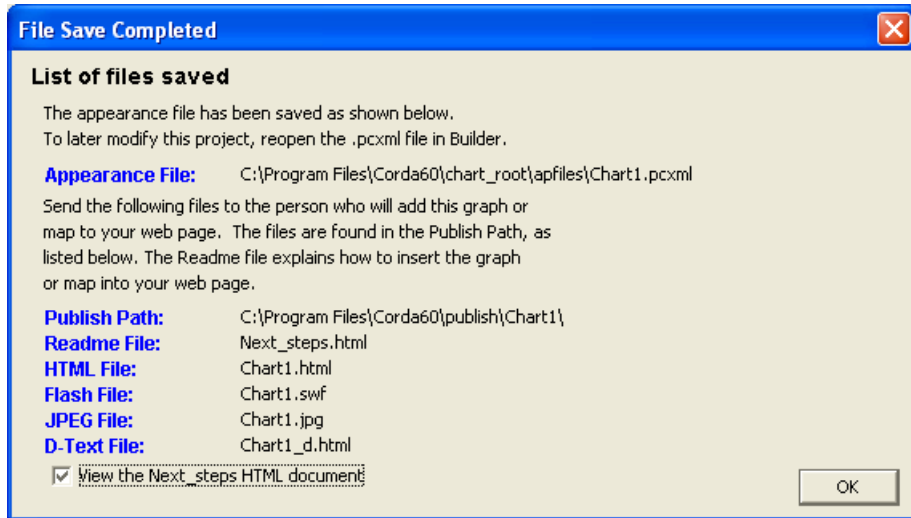


- 2 Select the desired output options and click **Next**.

Specify the image types to create, and change which image type appears in the Web page. For more information about these output features, see [“Output Features” on page 13-7](#).

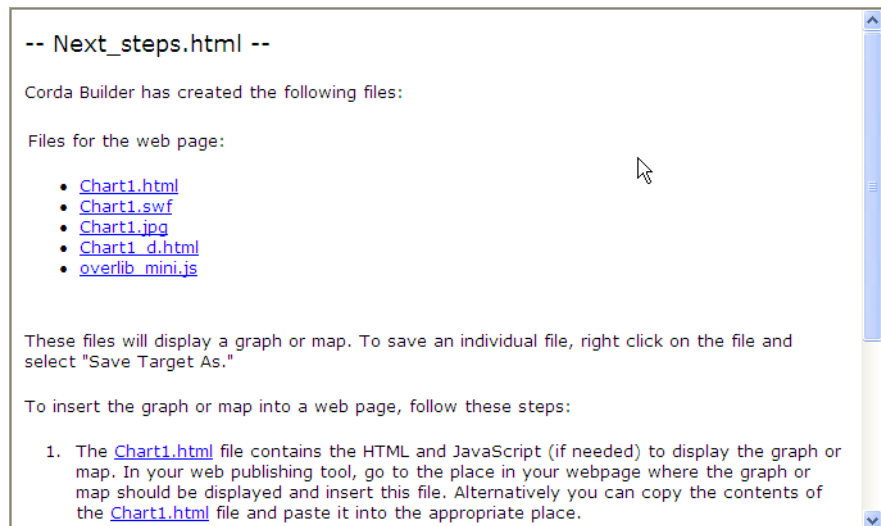
- 3 In the **Save File As** dialog, specify a name for the project file and click **Save**.

Corda Builder creates the selected image files and the sample Web page.



4 **Click OK to complete the publishing process.**

Corda Builder launches the system's default Web browser and loads a page with links to each image file and sample Web page.



- **PUBLISHING THE PROJECT**
- *Publishing a Project*
-
-

5 To view the published Web page, select the HTML link.

To learn how to access the image files that were output during the publishing process, see [“Accessing Published Files” on page 13-5](#).

To learn how to incorporate the sample Web page into the Web site, see [“Publishing Static Images on the Web” on page 13-10](#).

ACCESSING PUBLISHED FILES

All of the files for a published project are saved in `<product_root>\Builder\publish`. Each is stored in a directory that bears the same name as the Image Template file. For example, if the Image Template file is named `Chart1.itxml`, the published project directory is `Chart1`.

The **User Preferences** dialog allows you to change the directory to which the project directories are saved.

To change the publish directory

- 1 **Select** `Edit > Preferences`.
- 2 **Select** the **Folders** tab.
- 3 **Change** the location specified in the **Publish directory** field.
- 4 **Click** **OK** to save the changes.

Access the published project directory by selecting `File > Published Projects`.

The following table shows the files that are saved with a project and the purpose of each. Not all of these files are created with every project—it depends on the options selected when the project was published.

File	Example Filename	Purpose
EMF File	Project*.emf	The graph or map image in EMF format.
EPS File	Project*.eps	The graph or map image in EPS format.
HTML File	Project*.html	A sample Web page displaying the graph or map, as well as the HTML data table.
DLink File	Project*_d.html	The descriptive text for the image.
JPEG File	Project*.jpg	The graph or map image in JPEG format.
Javascript Library	overlib_mini.js	A Javascript library necessary for the Web page to show the image in Flash or SVG format.
Next Steps File	Next_steps.html	Instructions on how to deploy the graph or map image on the web site.
PDF File	Project*.pdf	The graph or map image in PDF format.
PNG File	Project*.png	The graph or map image in PNG format.
Flash File	Project*.swf	The graph or map image in Flash format.
SVG File	Project*.svg	The graph or map image in SVG format.
TIFF File	Project*.tif	The graph or map image in TIFF format.

Use any of the images in this directory within other applications or documents. To deploy an image in a Web page, copy the sample Web page file to the Web site (in addition to the images the Web page displays), or copy the code from the sample page into another Web

- **PUBLISHING THE PROJECT**
- *Accessing Published Files*
-
-

page. Brief instructions for deploying the sample Web page can be found in the Next_Steps.html file.

A step-by-step tutorial for deploying the sample Web page can be found in [“Publishing Static Images on the Web”](#) on page 13-10.

OUTPUT FEATURES

When publishing a project, you can use several features to enhance project output, including adding descriptive text, adding data tables, and adding best image fallback code to ensure that an image always appears in the highest-resolution format supported by a Web browser.

- [Descriptive Text](#)
- [Data Tables](#)
- [Best Image Fallback](#)

DESCRIPTIVE TEXT

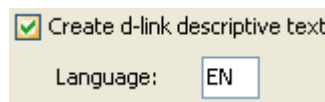
Generate descriptive text in a published project by enabling it in the **Publishing** dialog. When enabled, Corda Builder automatically inserts a [d](#) link next to the published Corda image. This link allows the visually impaired to browse to a descriptive text HTML page generated by Corda Builder.

Important: *To properly use descriptive text in a Web page, place the descriptive text file (Project_d.html) in the same directory as the Web page. Otherwise, you must change the HTML so that it looks for the descriptive text file in the proper location.*

For more information about descriptive text, see [“Descriptive Text” on page 11-5](#).

To enable descriptive text generation

- 1 **Select File > Publish.**
- 2 **Select Create d-link descriptive text.**



- 3 **Continue with the publishing process as normal.**

- PUBLISHING THE PROJECT
- Output Features
-
-

DATA TABLES

A Data table displays an HTML table of project data immediately below a graph or map. Data tables are automatically embedded in the Web page with the Corda Image.

Customize the appearance of this HTML table by modifying the table style. This and other aspects of data tables are explained in [“Data Tables” on page 11-11](#).

To enable data table generation

- 1 Select **File > Publish**.
- 2 Select **Create HTML Table**.



- 3 Continue with the publishing process as normal.

BEST IMAGE FALLBACK

Best image fallback enables a Web page that displays a Corda Image to output code in an alternate format if a user's Web browser does not support the default format. For example, if the default format is Flash and the Macromedia Flash Player is not installed on a user's Web browser, the image appears in JPEG format instead. If the default format is SVG and the Adobe® SVG Player is not installed, the image appears in Flash format as long as Flash is supported, or in JPEG format if it is not.

When enabled, this setting places Javascript that detects the image formats a Web browser can correctly display. It then dynamically displays the best supported image format.

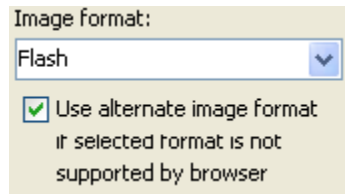
When Best Image Fallback is enabled, the necessary HTML and Javascript code is generated automatically when saving an Image template file. The code is inserted into the `ProjectX.html` file, where *ProjectX* is the name of the Corda Builder project.

Important: *To properly use alternative image formats when the user's browser does not support the default format, place the Flash, JPEG, and SVG (if selected) files created when the project was published in the same directory as the Web page into which the image is inserted. Otherwise, you need to change the HTML so that it looks for the alternate image files in the proper location.*

To enable best image fallback

Best image fallback is enabled by default. If necessary, enable it by doing the following:

- 1 **Select File > Publish.**



- 2 **Select Use alternate image format if selected format is not supported by browser.**
- 3 **Continue with the publishing process as normal.**

PUBLISHING THE PROJECT

Publishing Static Images on the Web

PUBLISHING STATIC IMAGES ON THE WEB

If a Web developer is posting a Corda Image to the Web for you, send the developer all of the files in the Published Project directory (see [“Accessing Published Files” on page 13-5](#)). With the help of the Next_Steps.html file, the developer should be able to publish the Corda image.

This section describes posting static Corda Images on the Web and assumes that you already know how to upload files to the Web server.

USING THE SAMPLE WEB PAGE

Corda Builder automatically generates a sample HTML file that displays the Corda Image as part of the publishing process. The simplest way to publish a graph or map to the Web is to upload the Corda Image’s project directory, including the HTML file, to the desired location on the Web server.

If desired, rename the HTML file, but don’t rename any other files. If desired, edit the project’s HTML file to include a title or any other information that is relevant to the Corda image. When editing the HTML, be careful not to erase or modify any code that is already in the file.

Note: *The Next_Steps.html file, and PDF, EPS, EMF, or TIFF image formats generated during the publishing process, are not required to deploy the Corda Image into an existing Web page, and do not have to be copied.*

USING AN EXISTING WEB PAGE

Typically, Corda Images are incorporated into an existing Web page. To do this, use a Web publishing tool, like Microsoft FrontPage®, or cut and paste HTML using a text or HTML editor.

- [To insert a Corda image using Microsoft FrontPage](#)
- [To insert a Corda image using an HTML Editor](#)

Note: *The sample HTML page used in the following instructions, as well as the sample files, are in <product_root>\Resources\image_templates\.*

To insert a Corda image using Microsoft FrontPage

- 1 Copy the Corda Image files from the project directory to the directory for the destination Web page.

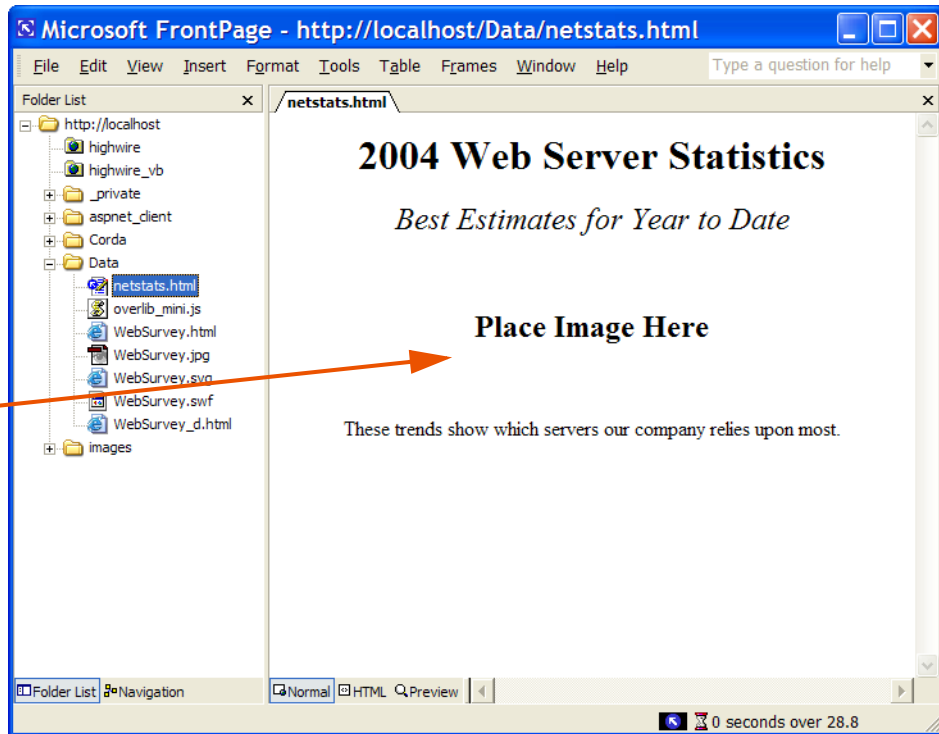
If you need to keep the image files elsewhere, see ["Placing the Image Files in a Different Directory than the Web Page"](#) on page 13-18.

Note: *The Next_Steps.html file, and PDF, EPS, EMF, or TIFF image formats generated during the publishing process, are not required to deploy the Corda Image into an existing Web page, and do not have to be copied.*

- 2 Open the destination Web page in Microsoft FrontPage and locate where you want to insert the Corda Image.

This is our Web page. We've opened it up in Microsoft FrontPage so we can insert the Corda image into it.

This is where we want to place the Corda image on our Web page.

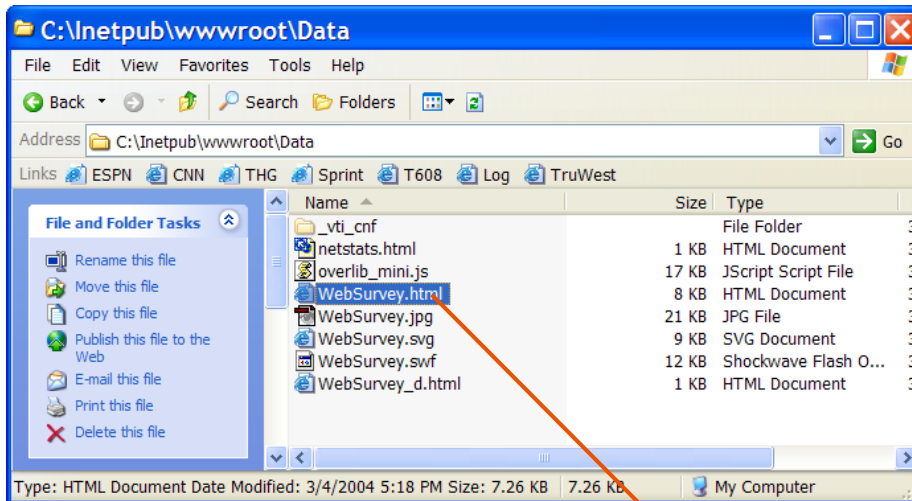


PUBLISHING THE PROJECT

Publishing Static Images on the Web

- Using a file manager such as Windows Explorer, locate the HTML file and drag it to the place on the Web page where you want to insert the Corda image.

Alternatively, place the cursor at the insertion location, select **Insert > File**, and then select the HTML file.



We've put all of our project files in this directory, which contains the document (netstats.htm) in which we want to embed our Corda Image.

We've dragged and dropped the WebSurvey.html file into our Web page. Now we can see it there, along with our [d](#) text link and data table.

Best Estimates for Year to Date

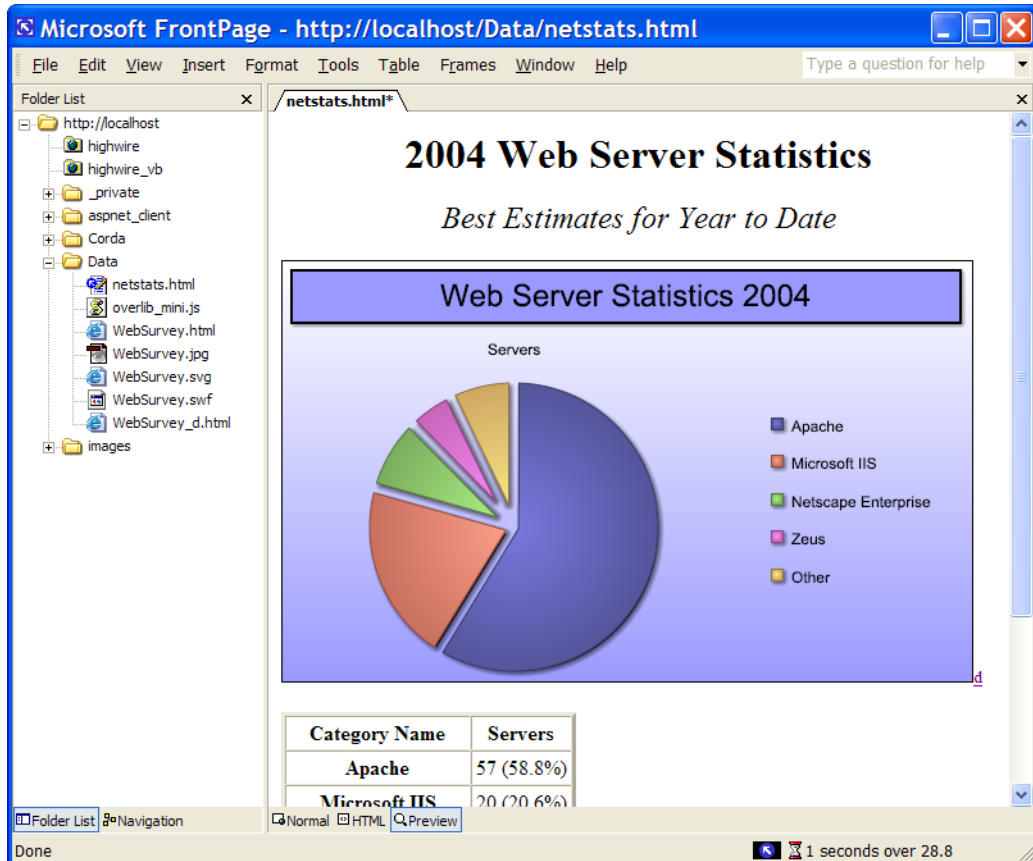
Category Name	Servers
Apache	57 (58.8%)
Microsoft IIS	20 (20.6%)
Netscape Enterprise	8 (8.2%)
Zeus	5 (5.2%)
Other	7 (7.2%)

These trends show which servers our company relies upon most.

1 seconds over 28.8

If successful, you see the HTML table and descriptive text link in the destination Web page, if these features were included in the Corda Image. The graph or map is not displayed because it is generated separately by Javascript code.

- 4 To view the image in the Web page, select the **Preview** tab at the bottom of the page.



- 5 Save the modified Web page.
- 6 Publish the modified Web page, along with any relevant image files (SVG, Flash, and/or JPEG), and the D-text file (if it exists) to the Web server.

Note: *Because of the way Microsoft FrontPage handles Javascript, you may encounter problems if you move the image around after you have inserted it. You may find it more convenient to remove the image entirely and then re-insert it in the appropriate location.*

- PUBLISHING THE PROJECT

- Publishing Static Images on the Web

To insert a Corda image using an HTML Editor

Note: Many HTML editors have the option to insert a file. You may be able to drag and drop, as you would with **Microsoft FrontPage**. You may also be able to insert a file using menu commands. These instructions assume that these options are not available.

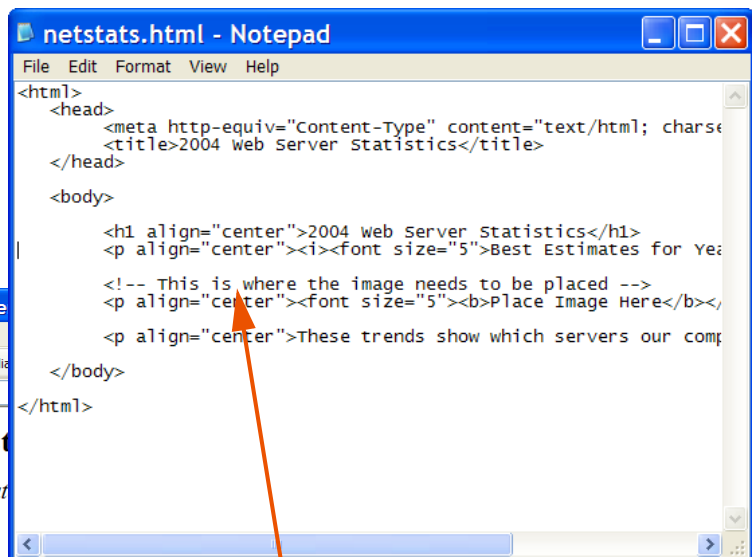
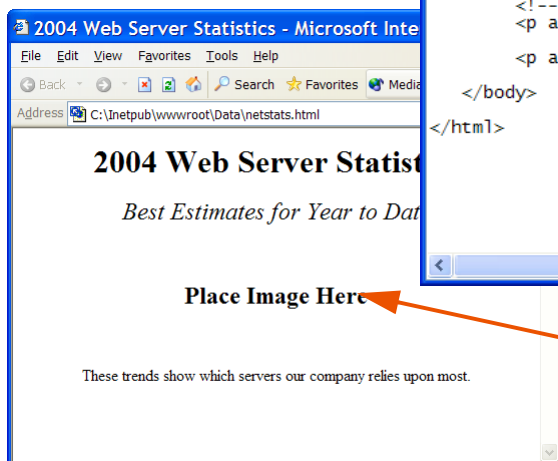
- 1 **Copy the Corda Image files from the project directory to the directory for the destination Web page.**

If you need to keep the image files elsewhere, see ["Placing the Image Files in a Different Directory than the Web Page"](#) on page 13-18.

Note: The *Next_Steps.html* file, and PDF, EPS, EMF, or TIFF image formats generated during the publishing process, are not required to deploy the Corda Image into an existing Web page, and do not have to be copied.

- 2 **With a text or HTML editor, open the Web page into which you want to insert the graph or map image.**
- 3 **Locate the place in the Web page's HTML code where you want to insert the Corda image.**

Here is our Web page, netstats.html, as it appears in **Internet Explorer** and in a text editor, before we've added our image.

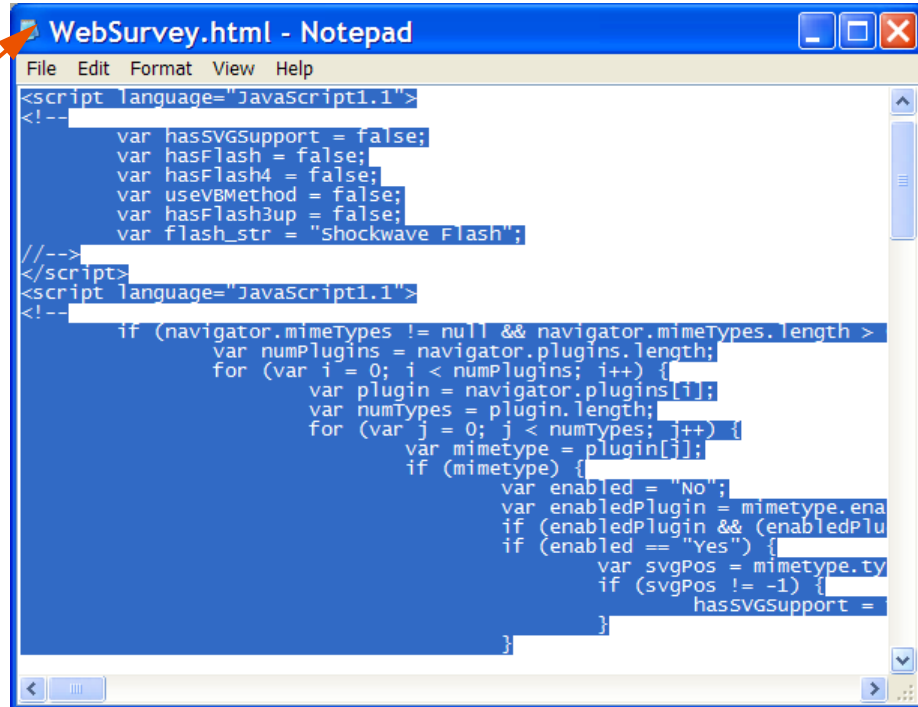


Place Image Here

The lower pointer shows where we want to insert the Corda image in our Web page. The upper pointer shows where we must insert the HTML code if we want to place it in this location.

- 4 With a text or HTML editor, open the Corda image's HTML page.
- 5 Select and copy the entire text of the Corda image HTML page.

This is the HTML file for our Corda image, WebSurvey.html, opened up in a different text editing window. We have selected the entire contents of the file. We should now copy what we have selected to the clipboard.



```
WebSurvey.html - Notepad
File Edit Format View Help
<script language="JavaScript1.1">
<!--
    var hasSVGSupport = false;
    var hasFlash = false;
    var hasFlash4 = false;
    var useVBMethod = false;
    var hasFlash3up = false;
    var flash_str = "Shockwave Flash";
//-->
</script>
<script language="JavaScript1.1">
<!--
    if (navigator.mimeTypes != null && navigator.mimeTypes.length >
        var numPlugins = navigator.plugins.length;
        for (var i = 0; i < numPlugins; i++) {
            var plugin = navigator.plugins[i];
            var numTypes = plugin.length;
            for (var j = 0; j < numTypes; j++) {
                var mimetype = plugin[j];
                if (mimetype) {
                    var enabled = "No";
                    var enabledPlugin = mimetype.enabledPlugin;
                    if (enabledPlugin && (enabledPlugin == "Yes")) {
                        if (enabled == "Yes") {
                            var svgPos = mimetype.svgPos;
                            if (svgPos != -1) {
                                hasSVGSupport = true;
                            }
                        }
                    }
                }
            }
        }
    }
-->
```

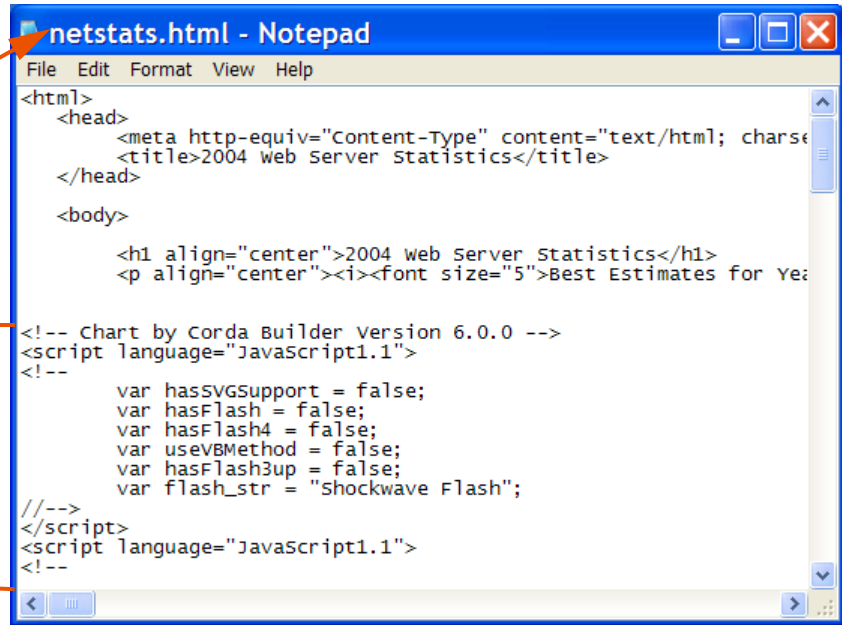
PUBLISHING THE PROJECT

Publishing Static Images on the Web

- 6 Paste the Corda Image HTML into the destination Web page at the insertion point you located in Step 2.

This is our Web page, netstats.html.

We've just pasted everything that we copied in the previous step into our Web page, at the location we determined in Step 3. (Note that the HTML we pasted extends beyond the visible portion of the window.)



```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charse
  <title>2004 web Server Statistics</title>
</head>
<body>

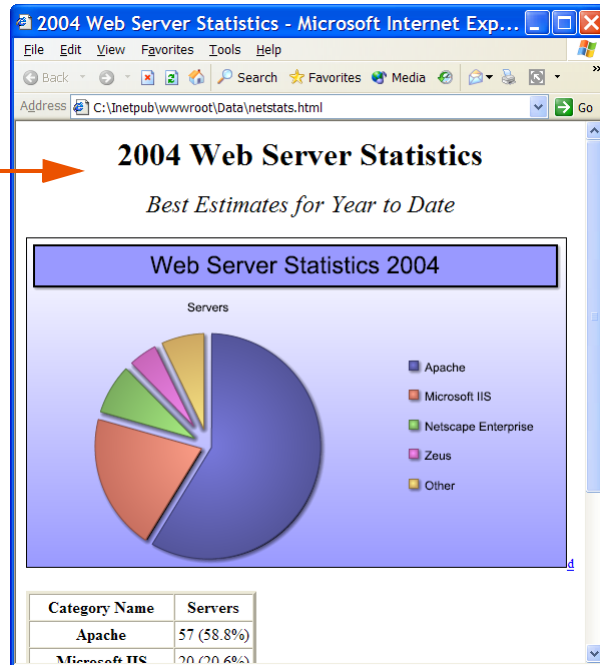
  <h1 align="center">2004 web Server Statistics</h1>
  <p align="center"><1><font size="5">Best Estimates for Yea

<!-- Chart by Corda Builder version 6.0.0 -->
<script language="JavaScript1.1">
<!--
  var hasSVGSupport = false;
  var hasFlash = false;
  var hasFlash4 = false;
  var useVBMethod = false;
  var hasFlash3up = false;
  var flash_str = "Shockwave Flash";
//-->
</script>
<script language="JavaScript1.1">
<!--
```

7 Save the modified Web page.

The Corda Image is now displayed in the destination Web page when you view it.

We can now view our Web page, netstats.html, with the Corda image in it.

**8 Upload the modified Web page, along with any relevant image files (SVG, Flash, and/or JPEG), and the D-text file (if it exists) to the Web server.**

- PUBLISHING THE PROJECT
- Publishing Static Images on the Web
-
-

PLACING THE IMAGE FILES IN A DIFFERENT DIRECTORY THAN THE WEB PAGE

Ideally, place all the Corda Image files in the same directory as the Web page that displays the Corda Image. If you need to place these files elsewhere, instruct Corda Builder to generate HTML code that looks for the image files in a different location. To do this, set the **URL Prefix** for the project.

To set the URL prefix

- 1 In Corda Builder, select **File > Publish**.
- 2 In the **URL Prefix** field, enter the relative or absolute path to the directory in which the Corda Image files should be saved.

Relative paths are relative to the location of the Web page that displays the Corda Image. For example, if the destination Web page is `http://www.sample.com`, and the **URL Prefix** is `images`, the Web page looks for all Corda Image files in the `http://www.sample.com/images` directory. If **URL Prefix** is set to `http://www.imagebox.com`, the Web page looks for all Corda Image files at `http://www.imagebox.com`.

Important: Once **URL Prefix** is set for a project, the published Corda Image isn't viewable until you copy all its related files (except for the HTML file) to the **URL Prefix** location setting. Include the D-text file, if necessary.

- 3 Continue with the publishing process as normal.

Alternatively, change the location of Corda Image files by modifying all references to Corda image files in the Web page's HTML to reflect their new location. For example, suppose you moved Corda Image files into a directory called `images`, and the following line appears in the Web page:

```

```

You would change it to read:

```

```


PUBLISHING DYNAMIC IMAGE TEMPLATE FILES WITH CORDA BUILDER'S SAMPLE CODE

When using Corda Builder to create dynamic Corda Images with Corda Server, Corda Builder can jump-start the Web deployment process by automatically generating Web code to embed a dynamic Corda image for you. Simply load or create an Image Template file, and then select the **Sample Code** button in Corda Builder's toolbar. From here, generate a complete example Web page for the Corda image.

Important: *Save the Image Template file before creating the sample Web page. If Corda Server is running on a different computer, save the Image Template file to the computer running Corda Server.*

Corda Builder's sample code interface allows you to generate example code for a variety of environments, including JSP, ASP, PERL scripts, PHP, ColdFusion pages, .NET applications, and just standard HTML. This code uses Corda Embedder to create and display the Corda image within the Web page. The interface also allows you to specify important variables, such as the image type and a server address.

This code works immediately when saved directly to the Web server, although you typically customize it further before publishing the Corda image. Either way, the sample code can save a lot of time. And, of course, it's a great learning tool.

Note: *A Corda Server administrator needs to install Corda Embedder for the example code to work. Information about installing Corda Embedder can be found in Chapter 5, "Using Corda Embedder," of the Corda 7 Developer Reference.*

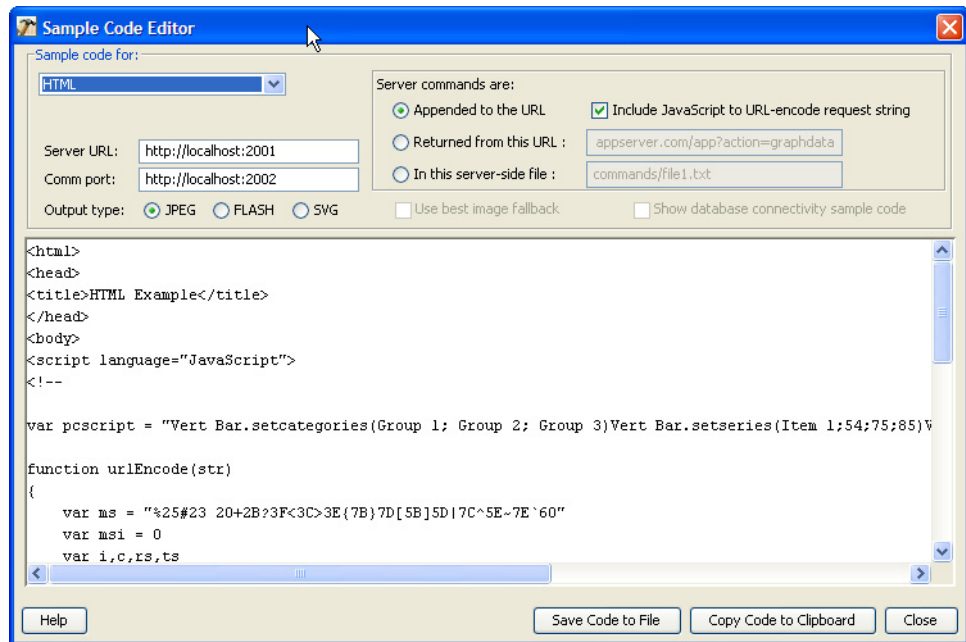
- [Sample Code Options](#)
- [Testing the Image Template File](#)

PUBLISHING THE PROJECT

Publishing Dynamic Image Template Files with Corda Builder's Sample Code

SAMPLE CODE OPTIONS

Several buttons and options are used in the **Sample Code** interface. This section explains each of them.



SERVER URL

Specifies the Corda Server address. If Corda Server is running locally, leave this value as `http://localhost:2001`. If you don't know where Corda Server is running, ask the Corda Server administrator or see ["Identifying Server Addresses"](#) on page 5-6 of the *Corda 7 Developer Reference*.

COMM PORT

Specifies the Corda Server's comm port address. If the Corda Server is running locally, leave this value as `http://localhost:2002`. If you don't know where Corda Server is running, ask the Corda Server administrator or see ["Identifying Server Addresses"](#) on page 5-6 of the *Corda 7 Developer Reference*.

OUTPUT TYPE

Specifies the output type for the Corda Image. Options include JPEG, FLASH, and SVG.

PUBLISHING THE PROJECT

Publishing Dynamic Image Template Files with Corda Builder's Sample Code

USE BEST IMAGE FALLBACK

Enables best image fallback, which detects the plug-ins a browser has and allows the Web browser to always show a Corda Image using the best possible image format supported by that Web browser. For more information, see [“Best Image Fallback” on page 8-3](#) of the *Corda 7 Developer Reference*.

SHOW DATABASE CONNECTIVITY SAMPLE CODE

Inserts sample code for database connectivity. This code refers to the **CordaSamples** database that is installed by default on most Windows systems. In fact, if you use it on a Windows system, the sample page/Web application actually connects to the **CordaSamples** database. However, most database programmers find it fairly easy to modify this code to connect to their own database.

To learn more about the **CordaSamples** database, see [Appendix C, “Corda 7 Sample Databases,”](#) in the *Corda 7 Install and Administration Guide*.

SELECT CODE FILE (FOR ASPNET SAMPLES)

Shows either the HTML source for the Web page (the .aspx file), or the .NET code behind the Web page (the .aspx.cs or .aspx.vb file).

SERVER COMMANDS (FOR HTML SAMPLES)

Specifies that these pages use server commands instead of Corda Embedder to request a Corda image. This allows you to specify the method used to send server commands to Corda Server.

Appended to the URL Server commands are sent in the same request used to get the image from Corda Server. This is the easiest way to do things, but it may not work with some Web browsers if the request grows longer than 512 characters.

When using this method, URL-encode the request so that the browser properly transmits it to Corda Server. Select **Include JavaScript to URL-encode request string** to include a Javascript function to do this encoding for you.

Returned from this URL Generates a server command file based on a server command file located on another Web server or generated by a Web application on another server. Specify the URL to the server command file in the field to the right.

In this server-side file Specifies the location of a server command file for this image on Corda Server. Specify the file name (relative to Corda Server's <document_root> directory) in the field to the right.

SAVE PREFERENCES

Saves the specified settings for **Server URL**, **Comm Port**, **Code type**, and **Image Type** as default Sample Code preferences.

PUBLISHING THE PROJECT

Publishing Dynamic Image Template Files with Corda Builder's Sample Code

SAVE CODE TO FILE

Saves the sample code displayed in the Sample Code window to a file.

COPY CODE TO CLIPBOARD

Copies the sample code displayed in the Sample Code window to the system clipboard.

SAVE SAMPLE DATA FILE(S) (FOR MAP IMAGES ONLY)

Saves a sample data table to the [CordaSamples](#) database. This sample data table is compatible with the map you are currently using. If the [CordaSamples](#) database is not installed, Corda Builder saves the data as a CSV file in `<product_root>\Resources\examples\`. See [“Show Database Connectivity Sample Code”](#) on page 13-21.

TESTING THE IMAGE TEMPLATE FILE

The following process tests an Image Template file with the example code generated for the JSP Corda Embedder. You must have Corda Server running locally for this to work as described.

To test the Image Template file with the Java Corda Embedder

- 1 **Start Corda Builder.**
- 2 **Load, or create, an Image Template file.**
- 3 **Select the [Sample Code](#) button from the tool bar.**
- 4 **In [Sample Code For](#), select [JSP \(Java Embedder\)](#).**
- 5 **Select [Save Code to File](#).**
- 6 **Specify a file name and save it.**

To serve this page from a Web server, save it to, or subsequently upload it to, a Web-enabled directory.

- 7 **Start Corda Server locally.**
- 8 **View the Web page which you saved in step 6 in a Web browser.**

If you don't see a Corda image, make sure Corda Server has been started and that it is running at `http://localhost:2001`. If it is running on a different machine, change the [Server URL](#) value in the sample code dialog, and upload the Image Template file to Corda Server.

You can apply these instructions to most Web application environments (ASP, Java, etc.) by saving the file to the appropriate Web application server.

For more information about embedding Corda images, see Chapter 5, [“Using Corda Embedder,”](#) in the *Corda 7 Developer Reference*.

PREVIEWING A CORDA IMAGE

Preview a Corda image with the default Web browser, by clicking the Preview button, or by selecting **File > Preview**.



This launches the system's default Web browser, which displays a Web page with the various Corda Image formats selected when the project was published. A blank image indicates the Web browser does not have the proper plug-in to view that image format.

Change which image types appear in the preview page by going to **Edit > Preferences > General** and selecting or deselecting options under **Preview Options**.

For more information on using alternative image formats, refer to the previous section, "[Descriptive Text](#)."

- **PUBLISHING THE PROJECT**
- *Where Do I Get the SVG or FLASH Plug-In?*
-
-

WHERE DO I GET THE SVG OR FLASH PLUG-IN?

When a Web browser first encounters a Flash or SVG image, it prompts the user to download the plug-in for that image type. Select **Yes** to automatically download and install the plug-in.

Alternatively, download the current plug-in from the following sources:

MACROMEDIA FLASH PLAYER http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_Version=ShockwaveFlash

ADOBE® SVG VIEWER
<http://www.adobe.com/svg/viewer/install/main.html>

HOW DO I PRINT A CORDA IMAGE?

Corda images are easy to print. Print a Web page that contains a Corda image just as you would any other Web page. To print just the Corda image, follow the instructions for the appropriate image type below.

These instructions assume you are looking at the Corda image in a Web browser, which means it has already been published to the Web, or you are previewing the project (see [“Previewing a Corda Image” on page 13-23](#)). Make sure you have the appropriate plug-in for the image format you want to print (see [“Where Do I Get the SVG or FLASH Plug-In?” on page 13-24](#)).

Alternatively, locate the appropriate image file for the project individually and open it in an appropriate editor (e.g., Adobe Photoshop® for JPEG or Macromedia Flash for Flash).

To print a flash image

- 1 Right-click the image and select **Print**.

To print an SVG image

- 1 Right-click the image and select **View SVG**.

A new Web page appears with only the Corda Image in it.

- 2 Select **File > Print**.

To print a JPEG image

- 1 (Internet Explorer) Right-click the image and select **Print**.
- 2 (Netscape Navigator) Right-click the image and select **View Image**.

A new Web page appears with only the graph or map in it.

- 3 Select **File > Print to print**.

To print a PDF image

The PDF plug-in has a print button in the top right corner. Simply click this button to print the image.

PUBLISHING THE PROJECT

Tips for Creating Good Image Template Files

TIPS FOR CREATING GOOD IMAGE TEMPLATE FILES

When Corda Builder is used to create dynamic Image Template files for use with Corda Server, the image displayed in Corda Builder is often very different from that generated by Corda Server.

Consider the following factors when creating a dynamic Image Template file:

- [Object Names](#)
- [Image Template File Dimensions](#)
- [Sample Data](#)
- [Colors and Color Themes](#)
- [Object Resizing and Repositioning](#)
- [Image Format](#)

OBJECT NAMES

To send dynamic data to and/or customize objects in an Image Template file, each object must have its own unique name, which is used to reference the object when sending PCScript or ITXML commands to Corda Server. For example, the PCScript command to modify the data inside a text box is

```
textbox.SetText(My Title)
```

In this case, the name of the text box object is `textbox`. If the textbox were named `title`, for instance, the command would be:

```
title.SetText(My Title)
```

Thus, it is important to know the names of objects and make sure there are no duplicate names. By default, Corda Builder assigns objects unique names. For example, the first textbox in a project is named `textbox`, with subsequent textboxes named `textbox2`, `textbox3`, and so on.

Map names are predefined because they are usually named after the country or area they represent (e.g., `US` for a map of the United States).

You may want to change these names to be more descriptive. For example, a pie graph that shows voting percentages might have a default name of `graph2`. A more suitable name for this graph might be `votes`.

The object name is stored in the `name` property, and can be changed from Object Properties. More information on this property is available in the [Corda 7 Object Reference](#).

TITLE TEXT BOX

One very important object is the title text box. The text box named `title` is always considered the title text box. When you display descriptive text, the contents of this text box is used to determine the title of the image.

For more information, see “Text Boxes” on page 8-5.

IMAGE TEMPLATE FILE DIMENSIONS

When creating an Image Template file, try to make the dimensions of the Image Template file exactly the same as the dimensions of the images you want to output. Changing the dimensions of the Image Template file dynamically can lead to reduced image quality or unexpected cropping or repositioning of objects.

To change Image Template file dimensions, either click and drag out a corner of the Image Template file’s border in the Corda Builder interface, or specify the project’s width and height in the `width` and `height` properties in [Object Properties](#).

DYNAMICALLY RESIZING THE IMAGE TEMPLATE FILE

Dynamically resize the Corda images through the `height` and `width` Corda Embedder attributes or the `@_HEIGHT` and `@_WIDTH` server commands. However, remember that resizing Corda images does not mean that the image is scaled. It simply enlarges or reduces the dimensions of the Corda images. Objects may be repositioned or cropped if they are outside of the Image Template file boundaries.

Alternatively, scale the Corda image with the `htmlHeight` and `htmlWidth` Corda Embedder attributes. This not recommended for JPEG or PNG image formats because image quality suffers.

SAMPLE DATA

It is difficult to anticipate the data set for a dynamic Corda Image. A graph that looks good with a small sample data set may look horrible with twenty or thirty categories and series. A map’s sample data may look good with the defined map ranges, but those same map ranges might not work with the dynamic data. To help minimize surprises with dynamic data, make the sample data set as similar as possible to the expected dynamic data set.

Change a graph’s sample data in [Properties > Data Editor](#).

Important: *Test Image template files with a variety of data sets to make sure they respond well to different types of data.*

PUBLISHING THE PROJECT

Tips for Creating Good Image Template Files

COLORS AND COLOR THEMES

Set colors for all of the data series, not just the ones displayed in Corda Builder. That way, if the user inputs more data series than you expected, you still have control over the colors in the graph. Set these by changing the **Series Colors** buttons on the top half of the **Graph Properties > Colors** dialog. Alternatively, you may want to provide the user with color themes (refer to [“Creating New Color Themes” on page 11-26](#)).

OBJECT RESIZING AND REPOSITIONING

Dynamic data can cause objects in Image Template files, such as text boxes, graphs or maps, and legends, to grow to unexpected sizes. To compensate for this, Corda Server can try to move or resize objects. For example, if a text box doubles in height, overlapping with a graph, Corda Server can shrink the graph to compensate.

Most of this functionality is turned on by default. However, you may not want this behavior. For example, if you have a text box that you want to always stay the same size, you may need to turn off its **Auto Size Adjustment** option.

The following topics affect object repositioning:

- [Global Repositioning Options](#)
- [Graph Resizing](#)
- [Legend Resizing](#)
- [Text Box Resizing](#)
- [Graph Scale Label Adjustments](#)
- [Aligned Objects](#)
- [Anchored Objects](#)

GLOBAL REPOSITIONING OPTIONS

Corda Builder provides options at the project level for managing automatic object repositioning.

Properties and attributes related to global repositioning options are available from the **Image Template > Layout** property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

GRAPH RESIZING

All graph components are placed within predefined boundaries, including text, tick marks, scales, and everything on the grid area. By default, a graph's data is scaled to fit within these boundaries. This means that as the scales and scale label text grow, the grid area may shrink

to compensate. To avoid potential problems with this, set the tick marks and scales to be outside of the graph boundaries so that the grid area always remains the same size.

The size of scale labels may also affect graph labels (see [“Graph Scale Label Adjustments” on page 13-29](#)).

Properties and attributes related to automatic graph resizing are available from the `Graph Settings` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

LEGEND RESIZING

A legend never shrinks in size. By default, it also never grows in size, even if it runs out of room to display legend items. If necessary, change this default behavior in the following ways:

Maximum Size: Specify horizontal and vertical maximums to which the legend can grow.

Add Additional Columns: Add more columns to the legend as needed.

Shrink Legend Font: Shrink the legend font down to a minimum point size that you define.

Truncate Legend Labels: Truncate the legend labels to the specified number of characters.

Properties and attributes related to legend resizing are available from the `Legend Settings > Layout` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

TEXT BOX RESIZING

By default, text boxes automatically shrink so that the text fits exactly within the box, including the specified text box margin. However, Corda Builder supports additional resizing options for text boxes.

Properties and attributes related to text box resizing are available from the `Textbox Settings > Layout` property in `Object Properties`. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

GRAPH SCALE LABEL ADJUSTMENTS

Depending on the number of categories in a graph, Corda Server may have difficulty displaying all of the scales labels. When this happens, Corda Server can make automatic adjustments to accommodate the scale labels.

Corda Server applies these adjustments by first making each adjustment individually. If that doesn't work, it applies combinations of adjustments until the scale labels don't overlap, or its options are exhausted.

PUBLISHING THE PROJECT

Tips for Creating Good Image Template Files

Properties and attributes related to automatic scale label adjustments are available from the **Value Scale > Labels** property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

Corda Builder also provides manual adjustment options for scale labels. Properties and attributes related to manual graph scale label adjustments are available from the **Image Template > Layout** property in **Object Properties**. More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

ALIGNED OBJECTS

When you align a group of objects, Corda Server shrinks or grows these objects in such a manner that the objects are always aligned with each other.

To align objects

- 1 Select an object.**
- 2 Select the other objects. These objects is aligned to the first object.**
- 3 Select **Alignment** from the menu bar or right-click the selected item to display the alignment options.**
- 4 Specify the desired alignment option.**
 - **Left.** Aligns the second object, and any subsequently selected objects, with the left edge of the first selected object.
 - **Center.** Aligns the second object, and any subsequently selected objects, with the horizontal center of the first selected object.
 - **Right.** Aligns the second object, and any subsequently selected objects, with the right edge of the first selected object.
 - **Top.** Aligns the second object, and any subsequently selected objects, with the top edge of the first selected object.
 - **Middle.** Aligns the second object, and any subsequently selected objects, with the vertical middle of the first selected object.
 - **Bottom.** Aligns the second object, and any subsequently selected objects, with the bottom edge of the first selected object.

Note: *Depending on how the objects have been anchored, the alignment may not be preserved if Corda Server has to enlarge or shrink the objects.*

ANCHORED OBJECTS

Anchors are used to control the position of text boxes, graphs or maps, and legends as they expand.

For example, if additional text is added to a text box, the anchor determines how (in what direction) the text box adjusts for the additional text. If the anchor is set to **top-left**, then as

the text box content grows, the text box expands down and to the right to accommodate the additional text.

Properties and attributes related to anchored objects are available from the [Common](#) property in [Object Properties](#). More information on these properties and attributes is available in the [Corda 7 Object Reference](#).

IMAGE FORMAT

To achieve the best image quality, use Macromedia Flash or SVG image formats. Consider the following when serving images other formats:

PDF AND EPS PDF and EPS images allow no interactivity. Drilldown effects, rollover data labels, transparency, and hover text are not shown in these image formats.

PNG AND JPEG Interactivity for PNG and JPEG images may not be supported in some browsers, or if you do not use [Corda Embedder](#). For more details, see “[Drilldown / Hover / Roll-Over Data Label Problems with PNG or JPEG Images](#)” on page 11-11 of the *Corda 7 Install and Administration Guide*.

- **PUBLISHING THE PROJECT**
- *Tips for Creating Good Image Template Files*
-
-

