

# PDF Xpansion SDK 15

## Reference



# TABLE OF CONTENTS

INTRODUCTION .....	6
System Requirements .....	7
Installation of SDK.....	7
Contents of SDK package.....	8
Reference SDK libraries in Your Projects.....	9
Replace Trial License.....	10
Update of SDK Files.....	10
SDK Samples.....	11
Redistribution of PDF Xpansion .....	11
PDF XPANSION SDK .....	13
Getting Started .....	13
Load SDK Library.....	14
SDK Entry Point.....	15
Using Document Viewer .....	16
Error Handling / Exceptions .....	16
Combine, Convert, Import and Export Documents .....	17
Create, Edit or Explore the Page Content.....	17
Use Metadata (XMP) in PDF Files .....	17
Electronic Invoicing.....	17
Document Viewer .....	18
Coordinate System and Unit of Measure .....	19
.NET Forms Control .....	19
.NET WPF Control .....	21
UWP Control .....	23
ActiveX Control.....	26
SX Namespace .....	28
IRefObject Interface / IDisposable Interface .....	29
ObjPtr Class (Smart Ptr).....	30
PDFXpansionSDK Class.....	31
IApplication Interface .....	32
IAppFactory Interface .....	33
IAppSettings Interface .....	38
IBaseDocument Interface .....	38
ITextDocument Interface.....	40
IImageDocument Interface.....	40
IFragmentDocument Interface.....	40

ICompositeDocument Interface .....	41
IAcquisitionDocument Interface .....	41
ISearchProvider Interface .....	41
IPrintProvider Interface.....	43
ISequentialStream Interface .....	46
ICryptoCert Interface.....	48
ICryptoKey Interface.....	48
Base Types.....	48
Enumerations.....	49
Structures .....	51
SX::Graphics Namespace .....	54
IRegion Interface .....	54
IBitmapData Data.....	54
SX::Viewer Namespace.....	57
IWinViewer Interface .....	57
IViewer Interface .....	58
IViewerCanvas .....	64
IViewerConfig.....	72
IViewableDocument .....	72
IViewerEvents Interface / Delegate.....	73
IViewerEvent Interface.....	74
IViewerEvent_PDF Interface.....	75
IViewerEvent_PDF_Annot Interface.....	75
IViewerEvent_XPS_Link Interface .....	75
IViewerEvent_PopupActivity Interface.....	76
IViewerCancelableEvent Interface.....	77
IViewerEvent_SetActivePage .....	77
IViewerEvent_ScrollCanvas Interface.....	77
IViewerEvent_MouseActivity Interface .....	78
IViewerEvent_KeyboardActivity Interface .....	78
IViewerEvent_ToolActivity Interface .....	80
IToolState Interface .....	80
Enumerations.....	81
SX::PDF Namespace .....	84
Coordinate System and Unit of Measure .....	84
IDocument Interface.....	85
IDocProperties Interface.....	91
IDocSecurity Interface.....	92
IStandardSecurity Interface .....	93

IPasswordSecurity Interface .....	94
IPages Interface.....	95
IPage Interface .....	98
IAnnots Interface.....	102
IAnnot Interface .....	105
IFormFields Interface.....	105
IFormField Interface.....	110
IStatusPDFa Interface .....	110
IMessagePDFa Interface .....	111
Enumerations.....	113
SX::XPS Namespace .....	116
IPackage Interface.....	116
SX::Content Namespace.....	117
Coordinate System and Unit of Measure .....	117
IRichContent Interface.....	117
IRichCollection Interface .....	121
IRichObject Interface .....	124
SX::XMP Namespace .....	127
IDocument Interface.....	127
Enumerations.....	129
SX::ElInvoice Namespace .....	130
PDF/A-3 based invoice.....	130
IInvoiceDocument Interface.....	131
IInvoiceData Interface .....	135
IInvoiceDescription Interface .....	140
IInvoiceTransaction Interface .....	142
IAgreement Interface.....	144
IDelivery Interface.....	148
ISettlement Interface .....	152
ILineItem Interface.....	158
INote Interface .....	160
ITimePeriod Interface.....	160
IDateTime Interface .....	161
IParty Interface.....	162
IDeliveryTerms Interface.....	165
IRefDoc Interface.....	165
IProcProject Interface.....	167
IPrice Interface .....	167
IQuantity Interface .....	167

IAmount Interface .....	167
IGlobalID Interface .....	168
IOrganization Interface .....	169
IContact Interface .....	169
IAddress Interface .....	171
ICommCont Interface.....	172
IOrgID Interface.....	172
ITaxID Interface .....	173
IAttachment Interface.....	174
IChainConsignment Interface.....	174
IChainEvent Interface .....	174
IPaymentMeans Interface.....	175
ITax Interface .....	175
IAllowanceCharge Interface.....	177
ILogisticsCharge Interface .....	178
IPaymentTerms Interface.....	178
IAccount Interface.....	178
IMonetarySummation Interface.....	178
IAdvancePayment Interface.....	182
ICurrencyExchange Interface.....	182
ILineDoc Interface .....	182
ILineProduct Interface.....	182
ILineAgreement Interface.....	183
ILineDelivery Interface.....	183
ILineSettlement Interface .....	183
Enumerations.....	183
TECHNICAL SUPPORT.....	185

# INTRODUCTION

This reference contains the documentation of the **PDF Xpansion SDK**: interfaces, properties and methods, structures and types declared and used in the PDF Xpansion libraries.

Certain properties and methods that may be discoverable through SDK's introspection facilities are not documented here. Undocumented properties and methods should not be used. They are entirely unsupported and subject to change without notice at any time.

This reference is intended for developers familiar with C++, C# or VB.NET. The intended audience includes, but is not limited to these languages. Familiarity with the PDF and/or XPS file formats and the object models can be helpful.

Different development platforms in this Reference have been designated as:

- **Windows** – applications and services written in C++ as the classic Windows x86/x64 programs
- **DotNET** – .NET Framework applications (written in C# and VB.NET)
- **COM** – applications written in Delphi, C++ Builder and the languages which support OLE Automation
- **UWP** – applications for Universal Windows Platform

## DotNET:

PDF Xpansion SDK includes assemblies built in **Visual Studio 2019** for **.NET Framework 4.0 / 4.5 / 4.6 / 4.7 / 4.8**. If you need assemblies for other versions of .NET Framework, please request us.

**Important!** These assemblies require the redistributable files for **Visual Studio 2019** to be installed.

**COM:** All objects of this library implement IDispatch interface and can be used from many scripting languages: JScript, VBScript, VBA, etc.

The library provides the document viewer as a separate ActiveX module.

# System Requirements

PDF Xpansion SDK can be used without any functional restrictions and third-party components on following Windows versions:

- Windows 10 and Windows Server 2016
- Windows 8.1 and Windows Server 2012 R2\*
- Windows 8 and Windows Server 2012\*

\*UWP part of PDF Xpansion SDK cannot be used under these OS versions.

## Installation of SDK

You may install and use PDF Xpansion SDK without setup - just download ZIP file and unpack the archive into any folder of your hard drive.

### DotNET:

PDF Xpansion SDK assemblies require the redistributable files for **Visual Studio 2019** to be installed.

**UWP:** You need to install PDF Xpansion Visual Studio Extension using VSIX Installer.

## Contents of SDK package

### Folder “Docs”

The folder contains SDK documentation.

### Folder “DevRes”

**UWP:** The folder contains PDF Xpansion Visual Studio Extension “**pdf-xpansion-uwp.vsix**” for the projects of UWP applications.

### Subfolder “include”

**Windows:** The folder contains header files for classic C++ applications.  
You don’t need these files on the other platforms.

### Subfolder “samples”

The folder contains multiple sample projects for different developer platforms. Please read the chapter “[Redistribution of PDF Xpansion](#)” before starting the compiled sample because you should perform this procedure for a sample application also as for any other.

### Folder “Redist”

The folder contains redistributable files of SDK, more detailed information about using these files you find in chapter “[Redistribution of PDF Xpansion](#)”.

**UWP:** You do not need the files from this folder within UWP applications because all necessary files are installed with PDF Xpansion Visual Studio Extension.



# Reference SDK libraries in Your Projects

## Windows:

Folder **DevRes\include** contains the header files for C++ desktop or server applications. You should include these files to your project before you start to use the SDK API.

**Important!** Please define **SX\_WINDESKTOP\_DLL** macro in your project or in your source files if you want use the [document viewer](#) as a Windows window or other native elements of Windows provided by PDF Xpansion SDK. Please define **SX\_USE\_OBJ\_PTR** macro in your project or in your source files if you want use the [ObjPtr template class](#).

## DotNET:

- 1) Add new reference **pdf-xpansion-net4X-x86.dll** or **pdf-xpansion-net4X-x64.dll** from **redist** folder of PDF Xpansion SDK to your project before you start to use the SDK API.
  - 2) Add new reference **pdf-xpansion-wpf4X-x86.dll** or **pdf-xpansion-wpf4X-x64.dll** from **redist** folder of PDF Xpansion SDK to your project before you start to use **WPF control**.
- PDF Xpansion SDK assemblies require the redistributable files for **Visual Studio 2019** to be installed.

## COM:

- 1) Register **pdf-xpansion-com-x86.dll** or **pdf-xpansion-com-x64.dll** using regsvr32 utility before you start to use the SDK API or you can use library without registration ([see detailed information here](#)).
- 2) Register **pdf-xpansion-ctl-x86.dll** or **pdf-xpansion-ctl-x64.dll** using regsvr32 utility if you need to use an **ActiveX control**.
- 3) Use **pdf-xpansion-vba-x86.dll** or **pdf-xpansion-vba-x64.dll** for VBA projects
- 4) Embarcadero developers must import necessary components in the project – use **“Component\Import Component”** function of Embarcadero environment.

## UWP:

- 1) Add new reference **PDF Xpansion SDK** to your project before you start to use the SDK API. You must [install appropriate Visual Studio Extension package](#) if you can't find this reference in the list of available references.
- 2) If you want use the document viewer, please add **PDFXpansionControl** to your app. To add a control in XAML of your app, double-click it in the Toolbox under **PDF Xpansion SDK** category.

## Replace Trial License

After you purchase the PDF Xpansion SDK you get your corporate license file **pdf-xpansion.license** and you will need to overwrite trial license file in order to replace trial license by your own. If you have tried the SDK in several projects, you must overwrite the license file in these projects also.

**Note!** Simultaneously with license file you should replace the trial license key in your sources because every license has own license key, you cannot authorize your license with trial key and vice-versa.

### UWP:

On this platform, you must overwrite a license file in location where SDK Extension was installed because extension package installs trial license. Usually you can find installed Visual Studio Extension files (and trial license file **pdf-xpansion.license**) in the subdirectories of directories

**...\Microsoft SDKs\Windows Kits\10\ExtensionSDKs\pdf-xpansion-uwp**

**Important!** You should overwrite trial license every time after [update](#) of SDK files and reinstallation of Visual Studio Extension because extension package contains trial license always.

## Update of SDK Files

We publish systematic releases of SDK with new functions and fixed bugs. We recommend that you [download an actual version of SDK](#) regularly (at least every quarter) and simply unpack the archive into the same folder (overwrite old SDK files).

Also you need to overwrite all used [redistributable SDK files](#) in your products with actual files, excluding license file **pdf-xpansion.license** because SDK package contains trial license always.

### COM:

You must repeat COM dll registration and import of components ([see detailed instruction here](#)) after this operation.

### UWP:

You must reinstall [used Visual Studio Extensions](#) after every SDK update.

Simply open the actual VSIX file.

**Important!** You should [replace trial license](#) after this operation.

# SDK Samples

Folder “DevRes\samples” contains multiple sample projects for different developer platforms. Please read the chapter “[Redistribution of PDF Xpansion](#)” before starting the compiled sample because **you should perform this procedure for a sample application also as for any other**.

## Redistribution of PDF Xpansion

Files that need to be redistributed along with the client application that uses PDF Xpansion SDK are placed in the **Redist** folder of SDK. Some files have to be always redistributed, other only in some specific cases.

You should copy all necessary redistributable files to one directory – application folder or subfolder, usually it is **folder with compiled executable files** of your application or our sample. More detailed information about every file see in table below.

### DotNET:

**Important!** Only the .NET references will be copied automatically by Visual Studio to your application folder. **All other required redistributable SDK files you must copy yourself, see table below.**

Please note that assemblies require the redistributable files for **Visual Studio 2019** to be installed on the target computers.

### COM:

You must register all redistributed COM dlls on the client PCs before use.

**Important!** The component **pdf-xpansion-com-x86.dll** (or **pdf-xpansion-com-x64.dll**) can be used without registration if you can call C-compatible entry point within your projects ([see detailed information here](#)).

**UWP:** At this platform the necessary redistributable files will be automatically included in your app package. You do not need to redistribute any other SDK files.

<b>pdf-xpansion-windesktop-x86.dll</b> <b>pdf-xpansion-windesktop-x64.dll</b>	The main library of SDK, <b>it should be redistributed with all applications</b> (including .NET and COM, excluding UWP applications). If you build only x86 or x64 binaries in your project, you may redistribute accordingly x86 or 64 libraries only.
<b>pdf-xpansion.license</b>	The license file of the SDK library, <b>it should be redistributed with all applications</b> . Note! After you have purchased SDK license you will get your corporate license file and you will need to <a href="#">replace this file</a> .
<b>pdf-xpansion.pds</b>	The resources store for the SDK library, <b>it should be redistributed with all applications</b> .
pdf-xpansion-net4X-x86.dll pdf-xpansion-net4X-x64.dll	The assemblies for use in .NET applications. You should redistribute its with .NET applications only. You need to redistribute assemblies for used .NET Framework version and processor architecture only. You may place these assemblies in Global Assembly Cache (GAC) also, but you must use <a href="#">EntryPointEx</a> instead of <a href="#">EntryPoint</a> in this case.
pdf-xpansion-wpf4X-x86.dll pdf-xpansion-wpf4X-x64.dll	The assemblies with the document viewer as a WPF Control. It is the same redistribution rules for these assemblies, as for main SDK assemblies.
pdf-xpansion-com-x86.dll pdf-xpansion-com-x64.dll	The COM-module for Delphi, Builder and the languages that support OLE interfaces.
pdf-xpansion-ctl-x86.dll pdf-xpansion-ctl-x64.dll	The ActiveX-module for Delphi, Builder and other languages that support embedding of ActiveX.
pdf-xpansion-cjk.pds	This store contains resources for support of CJK languages.
pdf-xpansion-en.pds pdf-xpansion-de.pds	This store contains localizable standard stamps. If you need other languages, please contact us.

# PDF XPANSION SDK

## Getting Started

1. Please [install PDF Xpansion SDK](#) at your computer.
2. If you already purchased SDK license, please [replace the trial license with your own](#), in other case you can test PDF Xpansion SDK with trial license. The trial license has not any functional restrictions, but applies DEMO watermark on the document pages. In addition, it has time restriction: you can use the trial license 3 months, after this time you need to download actual version of SDK and continue your tests.
3. Now you may try to compile [our samples](#) or you may try to integrate PDF Xpansion SDK in your project, in a second case you need [to add the SDK references in project](#).
4. Please read the chapter [SDK Entry Point](#) before integrate PDF Xpansion SDK in your project. Also read the chapter [Using Document Viewer](#) if you want integrate document viewer in your application.
5. Please read the chapter [SX::IRefObject Interface / IDisposable Interface](#) (for C++ and .NET applications only) about controlling lifetime of SDK objects and management of used resources.
6. Please read the chapter [Error Handling / Exceptions](#) about processing of SDK errors in your projects.
7. Please read the chapter [Redistribution of PDF Xpansion](#) before starting the compiled sample or your application because you should perform this procedure for any application that uses PDF Xpansion SDK.

## Load SDK Library

On most platforms the PDF Xpansion SDK library will be loaded to your application automatically and you can get [entry point of SDK library](#) anytime.

### Windows:

Before you can use the library in your application, you must load it from [pdf-xpansion-windesktop-x86.dll](#) or [pdf-xpansion-windesktop-x86.dll](#) module using [LoadLibrary](#) or [LoadLibraryEx](#) Windows function.

After successful loading of library, you can use module handle to get [entry point of SDK library](#) anytime. See our sample as an example.

### COM:

Optionally you can use the library without registering at the client PC. In this case, you need to load it from [pdf-xpansion-com-x86.dll](#) or [pdf-xpansion-com-x64.dll](#) module using [LoadLibrary](#) or [LoadLibraryEx](#) Windows function, if your environment allows this operation.

After successful loading of library, you can use module handle to get [entry point of SDK library](#) anytime.

# SDK Entry Point

We recommend you to get entry point of SDK library during the initialize phase of your application. See our samples as an example.

**Important!** PDF Xpansion SDK need to be authorized before you can use it. Please call [IApplication::Authorize](#) method during the initialize phase of your application.

## Windows:

At first, you should get address of **EntryPoint** function using [GetProcAddress](#) system function and [handle of loaded library](#). Than you can cast a retrieved address to **SX::LibraryEntryPoint** and call **EntryPoint** function - as a result you get [SX::IApplication interface](#).

## DotNET:

you should use **EntryPoint** method of the [SX.NET.PDFXpansionSDK](#) class to get [SX.NET. IApplication interface](#). If you use [Forms Control](#) or [WPF Control](#) in your project, you can use **Application** property as an entry point of SDK.

If you place SDK assemblies in Global Assembly Cache (GAC), you must use **EntryPointEx** method of the [SX.NET.PDFXpansionSDK](#) with additional parameter.

## COM:

you should create an object of [PDFXpansionSDK](#) class (ID is **SX.PDFXpansionSDK.15**) and use **EntryPoint** method of this object to get [IApplication interface](#).

**Note!** If you have not registered COM dll in registry but you can [load it dynamically](#), you can get address of **EntryPoint** function using [GetProcAddress](#) system function and [handle of loaded library](#). Than you can cast a retrieved address to **SX::LibraryEntryPoint** and call **EntryPoint** function - as a result you get [IApplication interface](#).

If you use our [ActiveX](#) in your project, you can use **Application** property as an entry point of SDK.

## UWP:

You should use **EntryPoint** method of [SX.UWP.PDFXpansionSDK](#) class to get [SX.UWP.Application class](#).

## Using Document Viewer

The PDF Xpansion SDK provides several platform-oriented implementations of document viewer. Using these implementations, you can easily and quickly integrate [a power viewer \(and/or annotator, editor\)](#) in your application.

## Error Handling / Exceptions

PDF Xpansion SDK uses the mechanism of exceptions and throws the exception of specific type which represent errors that occur during application execution. Please catch and process these exceptions everywhere you use PDF Xpansion SDK.

### Windows:

SDK throws C++ exceptions of the **SX::IException** type. The **ErrorCode** property of this interface corresponds to the [ErrorCode](#) enumeration.

### DotNET:

SDK throws exceptions of the **SX.NET.Exception** type which inherits [System.Exception](#) class. The **ErrorCode** property of this interface corresponds to the [ErrorCode](#) enumeration.

### COM:

SDK do not throw exceptions on this platform because traditionally uses **HRESULT** return value. All methods on this platform return a value of the type **HRESULT** which corresponds to the [ErrorCode](#) enumeration. Please find more information about [error handling in COM](#).

### UWP:

SDK throws exceptions of the [Platform.COMException](#) type. The **HResult** property of an exception object corresponds to the [ErrorCode](#) enumeration.



## Combine, Convert, Import and Export Documents

The PDF Xpansion SDK provides [a simple and powerful possibility to combine PDF and/or XPS documents](#). It combines not only the pages, but the hyperlinks, bookmarks and outline of document. Within the PDF format can be correctly combined the annotations of document, form fields and layers. Using this functionality one document can be split up into multiple documents (for example, one page – one separate document).

The library makes quick and high-quality conversion from PDF to XPS or OXPS format and backward conversion. Using the library many other formats can be imported to the PDF:

- plain text can be formatted and layouted at the pages in PDF document
- raster or SVG image can be imported as new page or placed on the existing page
- GDI or GDI+ metafile can be imported as new page or placed on the existing page

Using the library the pages of PDF or XPS document can be rendered and exported to the raster images, also the plain text can be extracted from such documents.

The library [makes conform PDF/A \(versions 1, 2 and 3\)](#) files from PDF files or images.

## Create, Edit or Explore the Page Content

The PDF Xpansion SDK provides direct access to the page content in PDF and/or XPS documents. See details about the [Rich Content API](#) within SDK. The viewer provides interactive tool for creating or editing page content by end user.

## Use Metadata (XMP) in PDF Files

You can get metadata of PDF document or edit such metadata using [XMP API](#) which is part of PDF Xpansion SDK.

## Electronic Invoicing

The PDF Xpansion SDK provides complete functionality for processing of electronic invoices, [see more detailed information here](#).

# Document Viewer

The PDF Xpansion SDK provides several platform-oriented implementations of document viewer. Using these implementations, you can easily and quickly integrate power viewer and editor of **PDF**, **XPS**, **ZUGFeRD** and other documents in your application. Also, you can display the documents of own format using customization possibilities of viewer.

The viewer implements a wide spectrum of traditional functions: different layouts, zooming, scrolling, etc. It has many tools for processing content: text marking, creating and editing of annotations (different types), page content editor.

The viewer uses modern and high-performance Windows API: **Direct2D** for displaying of documents. Direct2D takes advantage of hardware acceleration via the graphics processing unit and can minimize CPU usage and utilize hardware rendering on a graphics card.

The viewer processes all activity of end-user, including keyboard, mouse, touch and pen inputs. Using [event mechanism](#) provided by the viewer, your application could extend standard functionality of viewer and implement the own tools for interactive processing of PDF or other documents.

## Windows:

On this platform you can use viewer as a classic Windows window object represented by [SX::Viewer::WinViewer interface](#). This viewer displays a document and processes end-user activity.

As an alternative, you can use viewer object without window ([SX::Viewer::IViewer interface](#)) to manage viewport with layout of document in your own window.

## DotNET:

On this platform you can use viewer as an [Forms Control](#) or [WPF Control](#). These controls are derived from the appropriate control objects within .NET Framework and you can integrate it in your applications in the traditional way.

## COM:

On this platform you can use viewer as a classic [ActiveX](#) in your applications in the traditional way.

## UWP:

On this platform you can use viewer as a [custom control](#) that derived from the native user control object and you can integrate it in your applications in the traditional way.

## Coordinate System and Unit of Measure

The viewer coordinate system is based on the coordinate system of the display devices. The basic unit of measure is the device unit (typically, the pixel). The logical size of device unit within a viewer you can control using configuration interface of the viewer, especially DPI property. The x-coordinates increase to the right; y-coordinates increase from top to bottom.

The viewer operates with three similar coordinate systems:

- coordinate system of viewport
- coordinate system of viewed document canvas
- coordinate system of document page placed on the canvas

The origin of these coordinate systems coincides with the upper left corner of the viewport, canvas and page area accordingly. The offset of the viewport coordinate system relative to the canvas coordinate system is the scroll values and can be positive only. The placement of document page (page coordinate system) relative to the canvas coordinate system provides **IViewerCanvas::GetPageRect** method.

The pages in the PDF or XPS document have own native coordinate systems, these systems are different and do not coincide with the unified page coordinate system in the viewer. You can transform the page coordinates between unified and native systems using [PDF::IPage::CalcMatrix](#) or **XPS::IPage::CalcMatrix**.

## .NET Forms Control

This control is derived from [System::Windows::Forms::Control](#).

### Application Property

Type: [IApplication](#). Access: readonly.

The root of the PDF Xpansion API.

### Document Property

Type: [IViewableDocument](#). Access: full.

The property represents a document displayed in the viewer. Please note that you may not attach one IViewableDocument object in two or more viewers simultaneously. If you want display one document in the several viewers, you must request the necessary number of viewable document objects (one for every viewer).

## ToolMode Property

Type: [tool\\_mode](#). Access: full.

The property represents actual tool mode of the viewer.

**Important!** Before you set the value of this property to something else than "tool\_mode\_none" you must define [handler of viewer events](#) and associate it with viewer.

## Canvas Property

Type: [ViewerCanvas](#). Access: readonly.

The property provides the canvas of the viewer. Using this interface you can control layouting of the document in the viewer and navigate over the document.

## Config Property

Type: [ViewerConfig](#). Access: readonly.

The property provides the configuration settings of the viewer.

## EnableBarScroll Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of scrollbar events and scrolling of document in the viewer.

## EnableKeysScroll Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of keyboard events and scrolling of document in the viewer.

## EnableWheelScroll Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of mouse wheel events and scrolling of document in the viewer.

## Layout Method

The method recalculates layout of displayed document in the viewer if the document structure (number of pages, page size or page order) was changed.

## NewActPage Parameter

Type: [sx\\_uint](#).

The index of active page (see Canvas property above) after changing of document structure.

For example, before changes the active page was 7, than page 4 was deleted in document, after this change, you should call Realyout method and specify firts parameter as 6. In this case, the viewer recalculates the document layout but preserves (if possible) the active page and scrolling position.

If active page no more available, set this prameter to 0.

## AdoptPageOffset Parameter

Type: [sx\\_bool](#).

True if you want preserve visible part of active page after relayout.

## Invalidate Method

The method invalidates the viewport image.

## Pages Parameter

Type: [sx\\_bool](#).

True if content of pages was changed and page images in the viewer cache must be deleted.

## Pages Parameter

Type: [sx\\_bool](#).

True if PDF annotations in the document was changed.

## OnEvent Event

The viewer events.

Delegate: [IViewerEvents](#).

## .NET WPF Control

This control is derived from [System.Windows.Controls::UserControl](#).

## Application Property

Type: [IApplication](#). Access: readonly.

The root of the PDF Xpansion API.

## Document Property

Type: [IViewableDocument](#). Access: full.

The property represents a document displayed in the viewer. Please note that you may not attach one `IViewableDocument` object in two or more viewers simultaneously. If you want display one document in the several viewers, you must request the necessary number of viewable document objects (one for every viewer).

---

## ToolMode Property

Type: [tool\\_mode](#). Access: full.

The property represents actual tool mode of the viewer.

**Important!** Before you set the value of this property to something else than "tool\_mode\_none" you must define [handler of viewer events](#) and associate it with viewer.

---

## Canvas Property

Type: [IViewerCanvas](#). Access: readonly.

The property provides the canvas of the viewer. Using this interface you can control layouting of the document in the viewer and navigate over the document.

---

## Config Property

Type: [ViewerConfig](#). Access: readonly.

The property provides the configuration settings of the viewer.

---

## EnableBarScroll Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of scrollbar events and scrolling of document in the viewer.

---

## EnableKeysScroll Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of keyboard events and scrolling of document in the viewer.

---

## EnableWheelScroll Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of mouse wheel events and scrolling of document in the viewer.

---

## Relayout Method

The method recalculates layout of displayed document in the viewer if the document structure (number of

pages, page size or page order) was changed.

## NewActPage Parameter

Type: [sx\\_uint](#).

The index of active page (see Canvas property above) after changing of document structure.

For example, before changes the active page was 7, than page 4 was deleted in document, after this change, you should call Realyout method and specify firts parameter as 6. In this case, the viewer recalculates the document layout but preserves (if possible) the active page and scrolling position.

If active page no more available, set this prameter to 0.

## AdoptPageOffset Parameter

Type: [sx\\_bool](#).

True if you want preserve visible part of active page after relayout.

## Invalidate Method

The method invalidates the viewport image.

## Pages Parameter

Type: [sx\\_bool](#).

True if content of pages was changed and page images in the viewer cache must be deleted.

## Pages Parameter

Type: [sx\\_bool](#).

True if PDF annotations in the document was changed.

## OnEvent Event

The viewer events.

Delegate: [IViewerEvents](#).

## UWP Control

This control is derived from [Windows.UI.Xaml.Controls::UserControl](#).

## Application Property

Type: [IApplication](#). Access: readonly.

The root of the PDF Xpansion API.

## Document Property

Type: [IViewableDocument](#). Access: full.

The property represents a document displayed in the viewer. Please note that you may not attach one IViewableDocument object in two or more viewers simultaneously. If you want display one document in the several viewers, you must request the necessary number of viewable document objects (one for every viewer).

## ToolMode Property

Type: [tool\\_mode](#). Access: full.

The property represents actual tool mode of the viewer.

**Important!** Before you set the value of this property to something else than "tool\_mode\_none" you must define [handler of viewer events](#) and associate it with viewer.

## Canvas Property

Type: [ViewerCanvas](#). Access: readonly.

The property provides the canvas of the viewer. Using this interface you can control layouting of the document in the viewer and navigate over the document.

## Config Property

Type: [ViewerConfig](#). Access: readonly.

The property provides the configuration settings of the viewer.

## EnableBarScroll Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of scrollbar events and scrolling of document in the viewer.

## EnableKeysScroll Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of keyboard events and scrolling of document in the viewer.

## EnableWheelScroll Property

Type: [sx\\_bool](#). Access: full.



The property enables processing of mouse wheel events and scrolling of document in the viewer.

## EnableNavigationButtons Property

Type: [sx\\_bool](#). Access: full.

The property enables the “page back” and “next page” buttons on the left and right edges of the viewer. The viewer hides the buttons automatically after three seconds of inactivity.

## SnapToPageNavigationButtons Property

Type: [sx\\_bool](#). Access: full.

If true, than navigation buttons (see above) navigate to the top edge of document page. If false, than navigation buttons navigate to the previous or next “screen page”.

## EnableZoomButtons Property

Type: [sx\\_bool](#). Access: full.

The property enables a zoom control in the bottom right hand corner of the viewer. The viewer hides the buttons automatically after three seconds of inactivity.

## EnablePageIndicator Property

Type: [sx\\_bool](#). Access: full.

The property enables the page info in the bottom left hand corner of the viewer. The viewer hides the buttons automatically after three seconds of inactivity.

## EnableDrawProgress Property

Type: [sx\\_bool](#). Access: full.

The property enables a draw progress in the the viewer until the redraw complete.

## Suspended Property

Type: [sx\\_bool](#). Access: full.

The property allow to break and block any interaction with document (f.e. long time redraw) if you need to access the document from other thread (f.e. from semantic zoom view). Set this property to false if you want continue displaying the document in the viewer.

## Relayout Method

The method recalculates layout of displayed document in the viewer if the document structure (number of pages, page size or page order) was changed.

## NewActPage Parameter

Type: [sx\\_uint](#).

The index of active page (see Canvas property above) after changing of document structure.

For example, before changes the active page was 7, than page 4 was deleted in document, after this change, you should call Realyout method and specify firts parameter as 6. In this case, the viewer recalculates the document layout but preserves (if possible) the active page and scrolling position.

If active page no more available, set this prameter to 0.

## AdoptPageOffset Parameter

Type: [sx\\_bool](#).

True if you want preserve visible part of active page after relayout.

---

## Invalidate Method

The method invalidates the viewport image.

## Pages Parameter

Type: [sx\\_bool](#).

True if content of pages was changed and page images in the viewer cache must be deleted.

## Pages Parameter

Type: [sx\\_bool](#).

True if PDF annotations in the document was changed.

---

## OnEvent Event

The viewer events.

Delegate: [IViewerEvents](#).

---

## ActiveX Control

This is classic ActiveX control.

---

## Application Property

Type: [IApplication](#). Access: readonly.

The root of the PDF Xpansion API.

---

## Viewer Property

Type: [IViewer](#). Access: readonly.

The multifunctional viewer object.

# SX Namespace

Namespace “**SX**” is a root namespace of PDF Xpansion SDK, it contains declarations of base types, common enumerations, structures and interfaces used in the SDK.

**DotNET:** For this platform, root namespace is “**SX.NET**”.

**COM:** This platform does not use any namespaces.

**UWP:** For this platform, root namespace is “**SX.UWP**”.

# IRefObject Interface / IDisposable Interface

## Windows:

**IRefObject** is fundamental interface at this platform, it implements object lifetime management through reference counting. Most interfaces of PDF Xpansion SDK are inherited, directly or indirectly, from this interface.

See description of **AddRef** and **Release** methods below for more detailed information.

We recommend you use smart pointers (instances of an [ObjPtr Class](#)) everywhere you use **IRefObject** based interfaces to save pointer in a safe manner.

## DotNET:

Many objects of PDF Xpansion SDK use unmanaged resources and allocate unmanaged memory. Therefore most SDK interfaces at this platform are inherited, directly or indirectly, from [System.IDisposable interface](#). When you finish using an object that inherits **IDisposable**, you can reclaim the used by this object unmanaged resources and memory immediately.

## AddRef Method

The method increments the reference count for an object. Call this method for every new copy of an interface pointer that you make. You do not need to call this method if you use instances of an [ObjPtr Class](#) everywhere you use **IRefObject** based interfaces.

### Important!

All methods or properties of PDF Xpansion SDK call **AddRef** on a pointer before return it, **so you must call Release method** when you no longer need to use an interface pointer returned by SDK.

Please, **do not call AddRef method** additionally before passing it as an in parameter to any method or by setting a property within PDF Xpansion SDK.

## Release Method

Call this method when you no longer need to use an interface pointer returned by PDF Xpansion SDK or for pointer copies that you make. You do not need to call this method if you use instances of an [ObjPtr Class](#) everywhere you use **IRefObject** based interfaces.

## ObjPtr Class (Smart Ptr)

### Windows:

This template class is used at this platform only. Using the instances of this class provide some advantages over using a raw pointer of [IRefObject based interface](#):

- you can create instances tailored for a specific type of interface pointer (for any interface derived from **IRefObject**)
- call **AddRef** on the interface pointer received during an assignment operation
- provide different constructors to initialize a new instance through convenient mechanisms
- call **Release** for the encapsulated interface pointer when the class destructor executes

### Note!

Please define **SX\_USE\_OBJ\_PTR** macro in your project or in your source files if you want use this class.

# PDFXpansionSDK Class

The class provides an [entry point of the PDF Xpansion SDK](#) on the .NET, COM and UWP platforms.

## EntryPoint Method

This method returns [SX::IApplication interface](#).

### DotNET:

If you place .NET assemblies of PDF Xpansion SDK in one directory with other redistributable files (main libraries, license file, etc.), then you may use **EntryPoint** method to get [SX::IApplication interface](#).

If you place .NET assemblies of PDF Xpansion SDK in GAC, you must use **EntryPointEx** method instead of EntryPoint and specify a parameter - an absolute path to the main library, for example **C:\Program Files\Company\Product\sx\pdf-xpansion-windesktop-x86.dll**.

Both methods generate an exception [ErrorPathNotFound](#), if assembly cannot load the main library.

### COM:

If you place COM modules of PDF Xpansion SDK in one directory with other redistributable files (main libraries, license file, etc.), then you may use **EntryPoint** method to get [SX::IApplication interface](#).

If you place COM modules of PDF Xpansion SDK in other place, you must use **EntryPointEx** method instead of EntryPoint and specify a parameter - an absolute path to the main library, for example **C:\Program Files\Company\Product\sx\pdf-xpansion-windesktop-x86.dll**.

Both methods generate an error [ErrorPathNotFound](#), if assembly cannot load the main library.

# Application Interface

At the root of the PDF Xpansion API is the **IApplication** interface.

**Note!** PDF Xpansion SDK need to be authorized before you can use it. Please call [Authorize method](#).

## Version Property

Type: [sx\\_str](#). Access: readonly.

The property provides technical version of SDK.

## Authorized Property

Type: [sx\\_bool](#). Access: readonly.

The property provides state of SDK authorization. You can use authorized library only. See [Authorize method](#) for details.

## Factory Property

Type: [SX::IAppFactory](#). Access: readonly.

The property provides factory interface, you need it to instantiate any SDK objects.

## Settings Property

Type: [SX::IAppSettings](#). Access: readonly.

The property provides settings interface, you need it to set up SDK properties.

## Authorize Method

The method authorizes the holder of a license to access the permitted part API of PDF XpansionSDK. It must be called at least once, and is usually called only once, for application process that uses the library. Multiple calls to this method are allowed but not recommended as long as they do nothing after first successful call.

The method will attempt to locate the license file by looking for a file named “pdf-xpansion.license” in the same directory as the main library of SDK or by path specified as a second parameter.

## LicenseKey Parameter

Type: [sx\\_str](#).

The license key, you get it together with your own license, but unlike the license file, you should use license key content in your sources to call this method only. Please, don't redistribute key file with you application.



See our samples as an example (its use trial license key!).

## LicenseFilePath Parameter

Type: [sx\\_str](#).

If the redistributable license has a different filename and/or is not in the same directory as the main library of SDK, you should specify absolute path of license file here, otherwise this parameter must be null or empty string.

## Errors

The method produces an exception

- [ErrorPathNotFound](#) if can't find the license file.
- [ErrorAuthFailed](#) if used license key doesn't match the license or license file is corrupted.
- [ErrorAccessDenied](#) if can't find **pdf-xpansion.pds** file.
- [ErrorInvalidUse](#) if can't find **pdf-xpansion-windesktop-xXX.dll** file.
- [ErrorOutOfRange](#) if trial license is expired, please download actual version of PDF Xpansion SDK.

## IAppFactory Interface

The interface represents factory of SDK objects, it instantiates SDK objects permitted by license.

## CreateSequentialStream Method

The method creates new instance of memory based data stream. It returns [ISequentialStream interface](#).

## CreateReadStream Method

The method creates new instance of file based, data stream (read access only). It returns [ISequentialStream interface](#).

**UWP:** The method isn't available on this platform, but you can use ReadStreamAsync / ReadFileAsync methods of memory based [ISequentialStream](#) implementation.

## FilePath Parameter

Type: [sx\\_str](#).

The file path (can be filename, relative or absolute path).

## Errors

The method produces an exception [ErrorPathNotFound](#) if can't find or open the file.

## CreateWriteStream Method

The method creates new instance of file based, data stream (write access only, create always). It returns [ISequentialStream interface](#).

**UWP:** The method isn't available on this platform, but you can use WriteStreamAsync / WriteFileAsync methods of memory based [ISequentialStream](#) implementation.

### FilePath Parameter

Type: [sx\\_str](#).

The file path (can be filename, relative or absolute path).

### Errors

The method produces an exception [ErrorPathNotFound](#) if can't find or open the file.

## CreateRefStream Method

The method creates new instance of wrapper for the platform specific data stream. It returns [ISequentialStream interface](#). The wrapper object redirect all calls to the similar methods of base stream object.

**UWP:** The method isn't available on this platform, but you can use ReadStreamAsync / ReadFileAsync / WriteStreamAsync / WriteFileAsync methods of memory based [ISequentialStream](#) implementation.

### Stream Parameter

**Windows:** The [IStream](#) interface.

**DotNET:** The [System.IO.Stream](#) class.

## CreateCryptoCert Method

The method creates new instance of X.509 certificate. It returns [ICryptoCert interface](#).

## CreateCryptoKey Method

The method creates new instance of private key object. It returns [ICryptoKey interface](#).

## CreateRegion Method

The method creates new instance of graphic region object. It returns [Graphics::IRegion interface](#).

## Color Method

The method returns auxiliary Graphics::IColor interface.

## Matrix Method

The method returns auxiliary Graphics::IMatrix interface.

## CreateBitmapFromStream Method

The method creates new instance of bitmap object and loads bitmap data from the specified image stream (file). The method returns [Graphics::IBitmapData interface](#).

### Stream Parameter

Type: [ISequentialStream](#).

The image data stream.

## CreateBitmapFromSource Method

The method creates new instance of bitmap object and loads bitmap data from [IWICBitmapSource interface](#). The method returns [Graphics::IBitmapData interface](#).

**UWP:** The method isn't available on this platform.

**DotNET:** The method isn't available on this platform.

## CreateDrawingContext Method

The method creates new instance of SDK drawing context object. The method returns

Graphics::IDrawingContext interface.

**UWP:** The method isn't available on this platform.

## Drawer Parameter

Type: Graphics::Drawer.

The type of context.

---

## CreatePDFDocument Method

The method creates new instance of PDF document object. The method returns [PDF::IDocument](#) interface.

### Errors

The method produces an exception [ErrorAuthFailed](#) if your license does not permit use the PDF documents.

---

## CreateXPSPackage Method

The method creates new instance of XPS document object. The method returns [XPS::IPackage](#) interface.

### Errors

The method produces an exception [ErrorAuthFailed](#) if your license does not permit use the XPS documents.

---

## CreateTextDocument Method

The method creates new instance of document object with plain text. The method returns [ITextDocument](#) interface.

---

## CreateImageDocument Method

The method creates new instance of image document object. The method returns [IImageDocument](#) interface.

---

## CreateFragmentDocument Method

The method creates new instance of document object which is wrapper to other document. The method returns [IFragmentDocument](#) interface.

---

## CreateCompositeDocument Method

The method creates new instance of document object which is composition of several other documents. The method returns [ICompositeDocument](#) interface.

## CreateAcquisitionDocument Method

The method creates new instance of document object which is content provider for the acquisition devices (scanners, web cams, etc.). The method returns [IAcquisitionDocument](#) interface.

## CreateXMPDocument Method

The method creates new instance of XMP document which contains the [metadata](#). The method returns [XMP::IDocument](#) interface.

## CreateInvoiceDocument Method

The method creates new instance of PDF/A-3 based invoice object. The method returns [IInvoice::IInvoiceDocument](#) interface.

## CreateViewer Method

The method creates new instance of viewer object. The method returns [Viewer::IViewer](#) interface.

**Windows:** We recommend you [create window based viewer](#) instead of this object.

**UWP:** The method isn't available on this platform. Please read "[Using Document Viewer](#)" for detailed information about using viewer on this platform.

**DotNET:** The method isn't available on this platform. Please read "[Using Document Viewer](#)" for detailed information about using viewer on this platform.

## Errors

The method produces an exception [ErrorAuthFailed](#) if your license does not permit use the document viewer.

## CreateWinViewer Method

The method creates new instance of viewer window. The method returns [Viewer::IWinViewer interface](#).

**Windows:** You must define **SX\_WINDESKTOP\_DLL** macro in your project if you need to use window based viewer.

**UWP:** The method isn't available on this platform. Please read "[Using Document Viewer](#)" for detailed information about using viewer on this platform.

**DotNET:** The method isn't available on this platform. Please read "[Using Document Viewer](#)" for detailed information about using viewer on this platform.

## Rc Parameter

Type: [sx\\_rect](#).

The size and location of the window relative to the top-left corner of the parent window.

## Parent Parameter

Type: [sx\\_window](#) (typedef of HWND).

A handle to the parent or owner window of the window being created.

## Style Parameter

Type: [sx\\_flags](#), see also [Viewer::viewer\\_flags](#).

The optional styles of the [viewer window](#).

# IAppSettings Interface

The interface provides properties and options of PDF Xpansion SDK.

# IBaseDocument Interface

The interfaces declares common functionality for all documents supported by SDK.

**Windows:** This interface is derived from [IRefObject](#) at this platform. We recommend you use an instance of an [ObjPtr template class](#) where you can use an IBaseDocument pointer in a safe manner.

## GetViewableDocument Method

The method returns [Viewer::IViewableDocument interface](#) for using in the [viewer](#). Please note that you may not attach one instance of this interface in two or more viewers simultaneously. If you want display one document in the several viewers, you must request the necessary number of instances (one for every viewer).

## GetSearchProvider Method

The method provides search provider of the document. It returns [ISearchProvider interface](#).

## Errors

The method produces an exception [ErrorAuthFailed](#) if your license does not permit to search in the text of document.

## GetPrintProvider Method

The method provides print provider of the document, using this provider you can organize printing of document or build thumbnails for print preview. It returns [IPrintProvider](#) interface. Please note, you need separate provider objects for every print job because you can not print two or more times using one provider object. See description of JobName parameter for more information about using print provider object.

**Note!** On **Windows 7** and **Windows Server 2008 R2** must be installed [Platform Update](#) if you want print the documents using IPrintProvider interface.

**UWP:** The method hasn't any parameters on this platform.

### JobName Parameter

Type: [sx\\_str](#).

The print job name. If you want use print provider for print preview only, you should specify empty string or null here. If you call GetPrintProvider with non-empty job name, some virtual printers can show own UI (for example "Microsoft Print to PDF" asks file path for resulting PDF).

### PrinterName Parameter

Type: [sx\\_str](#).

The name of the printer, print server or printer object.

### DevMode Parameter

Type: [sx\\_byte\\*](#).

A pointer to a [DEVMODE](#) structure that the printer uses for setup, it is optional parameter, it can be null.

#### DotNET:

On this platform the type of parameter is "array<unsigned char>^". Please copy unmanaged data to a managed memory block, size of block must be at least size of DEVMODE.

### Output Parameter

Type: [ISequentialStream](#).

The print output stream, it is optional parameter, it can be null. Using this parameter, you can implement “print to file” functionality, if selected printer supports this possibility. The stream must be writable.

## Errors

The method produces an exception [ErrorAuthFailed](#) if your license does not permit to print the documents.

## Clear Method

The method completely resets the document - all pages and other document resources are removed.

## ITextDocument Interface

The interfaces provides interface of plain text document object implemented by SDK.

This interface is derived from [IBaseDocument](#) that allow you to load text document in the document viewer and/or combine it with PDF / XPS documents.

You can create new instance of document object using [IAppFactory](#) interface.

## IImageDocument Interface

The interfaces provides interface of image document object implemented by SDK.

This interface is derived from [IBaseDocument](#) that allow you to load image document in the document viewer and/or combine it with PDF / XPS documents.

You can create new instance of document object using [IAppFactory](#) interface.

## IFragmentDocument Interface

The interfaces provides interface of wrapper document object implemented by SDK. The wrapper references to other document, but you can define restricted page range or redefine page order of original document.

This interface is derived from [IBaseDocument](#) that allow you to load wrapper document in the document viewer and/or combine it with PDF / XPS documents.

You can create new instance of document object using [IAppFactory](#) interface.



## ICompositeDocument Interface

The interfaces provides interface of composite document object implemented by SDK. The document represents several documents as one document. The child documents can be of different types: text, image, PDF, XPS, etc.

This interface is derived from [IBaseDocument](#) that allow you to load composite document in the document viewer and/or combine it with PDF / XPS documents.

You can create new instance of document object using [IAppFactory](#) interface.

## IAcquisitionDocument Interface

The interfaces provides interface of document object implemented by SDK. The document is content provider for the acquisition devices (scanners, web cams, etc.). You can compose new document from paper sources or from web camera using this interface.

This interface is derived from [ICompositeDocument](#) that allow you to load obtained document in the document viewer and/or combine it with PDF / XPS documents.

You can create new instance of document object using [IAppFactory](#) interface.

## ISearchProvider Interface

The interface provides possibility search text in the document.

### Text Property

Type: [sx\\_str](#). Access: readonly.

The property retrieves actual text pattern.

### Page Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves page index of last occurrence. It can be 0xFFFFFFFF if pattern wasn't found.

### Selection Property

Type: [Graphics::IRegion](#). Access: readonly.

The property retrieves location of last occurrence at the page. The coordinates of region are in the

coordinate system of page.

## Reset Method

The method resets the search parameters. It doesn't search first occurrence, you must call SearchForward or SearchBack for start.

### Text Parameter

Type: [sx\\_str](#).

The text pattern to find.

### Page Parameter

Type: [sx\\_uint](#).

The index of start page.

### searchFlags Parameter

Type: [sx\\_flags](#), see also [search\\_opt](#) enumeration.

The parameter specifies the options of search. You can combine the enumeration values by using the bitwise OR operator.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## SearchForward Method

The method starts or continues search in forward direction. It returns true if text pattern was found or false if search reaches end of document. See Text, Page and Selection properties if next occurrence was found.

### StatusHandler Parameter

Type: [IStatusHandler](#).

The optional parameter sets a callback handler of search process. The [IStatusHandler](#) interface must implemented by application to get status events of search.

**UWP:** At these platforms the [IStatusHandler](#) delegate is declared instead of callback interface. Please define an event handler method with the same signature as the [IStatusHandler](#) delegate and add an instance to the **OnPage** event of the search provider.

## SearchBack Method

The method starts or continues search in backward direction. It returns true if text pattern was found or false if search reaches begin of document. See Text, Page and Selection properties if next occurrence was found.

### StatusHandler Parameter

Type: `IStatusHandler`.

The optional parameter sets a callback handler of search process. The `IStatusHandler` interface must implemented by application to get status events of search.

**UWP:** At these platforms the `IStatusHandler` delegate is declared instead of callback interface. Please define an event handler method with the same signature as the `IStatusHandler` delegate and add an instance to the **OnPage** event of the search provider.

## GetContext Method

The method retrieves text context of current occurrence. It returns [IStr](#).

### MaxLen Parameter

Type: [sx\\_uint](#).

Max length of context string.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## IPrintProvider Interface

The interface provides print possibility for the document and builds thumbnails for print preview. Please note, you need separate provider objects for every print job because you cannot print (call `StartPrint` method) two or more times using one provider object. You cannot call `StartPrint` method if you have created this provider for print preview purposes (with empty job name).

## PageSet Property

Type: `print_page_set`. Access: full.

The property defines whether to print all pages of the document

## PageRange Property

Type: [IStr](#). Access: full.

The property specifies the range of pages to print in the document (PageSet property must be print\_page\_set\_range). Separate numbers in a range by using a hyphen, and separate multiple pages or ranges by using commas.

## PageScale Property

Type: print\_page\_scale. Access: full.

The property defines scale mode: reduce, enlarge, or crop pages while printing.

## ReverseOrder Property

Type: [sx\\_bool](#). Access: full.

The property enables printing pages in reverse order. If page ranges were specified, the pages print opposite of the order in which they were entered.

## AutoRotate Property

Type: [sx\\_bool](#). Access: full.

The property adjusts the page orientation to match the paper (or printable area) orientation.

## Centered Property

Type: [sx\\_bool](#). Access: full.

The property specifies the alignment of the page with the center of the printable area.

## PrintAs Property

Type: print\_content\_as. Access: full.

The property specifies how to print every page.

## DrawOptions Property

Type: sx\_flags, see also draw\_opt enumeration.

The property specifies technical options of rendering.

## PDFAnnots Property

Type: sx\_flags, see also [PDF::annot\\_type](#) enumeration.

The property specifies the types of PDF annotations to be printed. You can combine the enumeration values by using the bitwise OR operator.

## StartPrint Method

The method prints specified pages of document to the selected printer. You cannot call this method two or more times. You cannot call this method at all, if you have created used provider object for print preview purposes (with empty job name).

**WinRT:** At this platform method is not available.

## StatusHandler Parameter

Type: [IPrintStatus](#).

The optional parameter sets a callback handler of search process. The [IStatusHandler](#) interface must implemented by application to get status events of printing.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## GetSheetsCount Method

The method retrieves number of sheets to be printed.

**WinRT:** At this platform method is not available.

## GetSheetThumbnail Method

The method retrieves thumbnail of specified sheet for preview. It returns [Graphics::IBitmapData interface](#).

**WinRT:** At this platform method is not available.

## Sheet Parameter

Type: [sx\\_uint](#).

The index of sheet.

## Width Parameter

Type: [sx\\_uint](#).

Thumbnail max width in pixels.

## Height Parameter

Type: [sx\\_uint](#).

Thumbnail max height in pixels.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## GetPrintDocumentSource Method

The method retrieves document source for a print task (see `PrintTaskRequest.CreatePrintTask`). It returns `Windows::Graphics::Printing::IPrintDocumentSource` interface.

**WinRT:** The method is available at this platform only.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## InvalidatePreview Method

The method forces the invalidation of print preview.

**WinRT:** The method is available at this platform only.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## ISequentialStream Interface

The interface supports simplified sequential access to stream data. The stream can be memory or file based. Also it could be a wrapper to other stream object.

**Windows:** This interface is derived from [IRefObject](#) at this platform.

---

## GetLength Method

The method return actual length in bytes of the stream.

---

## Reset Method

The method sets the stream pointer to the beginning of the stream data.

---

## Read Method

The method reads a sequence of bytes from the stream to the buffer and advances the pointer position

within the stream by the number of bytes read. It returns number of bytes copied to the buffer.

**UWP:** On this platform you can use the methods **ReadBytes** (synchronous read operation), **ReadStreamAsync** and **ReadFileAsync** (asynchronous read operation).

## Buf Parameter

Type: [sx\\_byte\\*](#).

The pointer to the memory buffer.

**DotNET:** On this platform the type of parameter is "array<unsigned char>^".

## Size Parameter

Type: [sx\\_uint](#).

The maximum number of bytes to be read from the stream.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Write Method

The method writes a sequence of bytes from the buffer to the stream and advances the pointer position within the stream by the number of bytes written. It returns number of bytes copied to the stream.

**UWP:** On this platform you can use the methods **WriteBytes** (synchronous write operation), **WriteStreamAsync** and **WriteFileAsync** (asynchronous write operation).

## Buf Parameter

Type: [sx\\_byte\\*](#).

The pointer to the memory buffer.

**DotNET:** On this platform the type of parameter is "array<unsigned char>^".

## Size Parameter

Type: [sx\\_uint](#).

The number of bytes must be writed to the stream.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## ICryptoCert Interface

The interface supports simplified sequential access to stream data.

**Windows:** This interface is derived from [IRefObject](#) at this platform.

## ICryptoKey Interface

The interface supports simplified sequential access to stream data.

**Windows:** This interface is derived from [IRefObject](#) at this platform.

## Base Types

PDF Xpansion SDK supports the type system that's defined by providing typedefs for the platform specific fundamental types:

<b>SX</b>	<b>Windows</b>	<b>WinRT</b>	<b>DotNET</b>
sx_bool	bool	bool	System::Boolean
sx_byte	unsigned char	Byte	System::Byte
sx_int	int	int32	System::Int32
sx_uint	unsigned int	uint32	System::UInt32
sx_short	unsigned short	uint16	System::Int16
sx_qword	unsigned __int64	uint64	System::UInt64
sx_float	float	Float32	System::Single
sx_double	double	float64	System::Double
sx_str ( <a href="#">IStr</a> )	const wchar_t*	Platform::String^	System::String^
sx_flags	sx_uint		



## IStr Interface

**Windows:** This interface is derived from [IRefObject](#) at this platform. The SDK API returns any dynamic strings as **IStr** interface, please do not forget to release dynamic strings when you no longer need to use it (see [IRefObject](#) description).

### str Property

Type: `sx_str`. Access: readonly.

The property retrieves the string content.

## Enumerations

### ErrorCode Enumeration

The members of this enumeration are the possible error codes for [exceptions generated by SDK libraries](#).

#### ErrorPathNotFound

The library cannot find the path specified.

#### ErrorAccessDenied

Access is denied.

#### ErrorOutOfMemory

Not enough memory is available to process this action.

#### ErrorOutOfSpace

The drive is full.

#### ErrorLibrary

Common library error.

#### ErrorSystem

Unknown system error.

#### ErrorAuthFailed

The library does not authorized or function does not permitted.

#### ErrorOutOfRange

Invalid index.

#### ErrorPermDenied

The action does not permitted by document permissions.

### **ErrorInvalidFormat**

Invalid data format.

### **ErrorNotImplemented**

The function does not implemented yet.

### **ErrorInvalidParam**

Invalid parameter value.

### **ErrorInvalidUse**

Invalid use of the function.

### **ErrorDocSecurity**

Inavlid use of encrypted document.

### **ErrorCryptoEngine**

Error occurred during crypto operation.

---

## **combine enumeration**

The members of this enumeration are the options for combine document operation. You can combine the enumeration values by using the bitwise OR operator.

### **combine\_named\_dests**

Copy all used named destinations to the target document.

### **combine\_outlines**

Copy outline structure to the target document.

### **combine\_form\_fields**

Copy form fields to the target document.

### **combine\_layers**

Copy layer objects to the target document.

### **combine\_links**

Copy links to the target document.

### **combine\_emb\_unk\_fonts**

Embed all non embedded and non Windows standard fonts.

## `combine_emb_win_fonts`

Embed all non embedded Windows standard fonts.

# Structures

## `sx_point`

Contains a set of two floating-point numbers that represent the point in a two-dimensional plane.

**DotNET:** The `System::Drawing::PointF` class used on this platform.

### `x`

The x-coordinate of the point.

### `y`

The y-coordinate of the point.

## `sx_size`

Contains a set of two floating-point numbers that specify a height and width.

**DotNET:** The `System::Drawing::SizeF` class used on this platform.

### `width`

The width component of the structure.

### `height`

The height component of the structure.

## `sx_rect`

Contains a set of four floating-point numbers that represent the location and size of a rectangle.

**DotNET:** The `System::Drawing::RectangleF` class used on this platform.

### `x`

The x-coordinate location of the left side of the rectangle.

## **y**

The y-coordinate location of the top side (or bottom side in PDF namespace, [see details here](#)) of the rectangle.

## **width**

The width of the rectangle.

## **height**

The height of the rectangle.

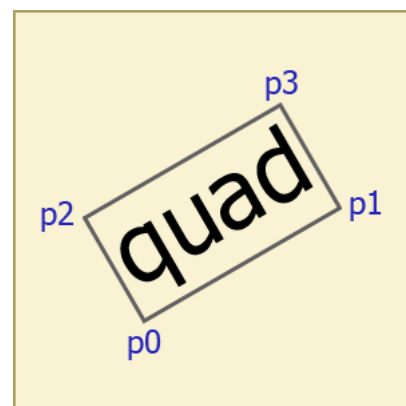
---

## **sx\_quad**

Contains a set of four `sx_point` structures that represent the parallelogram.

### **p0, p1, p2, p3**

The coordinates of four corners.



---

## **sx\_matrix**

Contains a 3-by-2 matrix.

### **m11**

The value in the first row and first column of the matrix.

### **m12**

The value in the first row and second column of the matrix.

### **m21**

The value in the second row and first column of the matrix.

### **m22**

The value in the second row and second column of the matrix.

### **dx**

The value of offset along the x-axis.

### **dy**

The value of offset along the y-axis.

---

## sx\_time

Contains the local date and time (for your time zone). The current time-zone settings used by SDK you can control over `IAppSettings::LocalTimeOffset` property.

**UWP:** The `Windows::Foundation::DateTime` class used on this platform.

**DotNET:** The `System::DateTime` class used on this platform.

### wYear

The year. The valid values for this member are 1601 through 30827.

### wMonth

The month. This member can be one of the following values.

### wDayOfWeek

The day of the week. This member can be one of the following values.

### wDay

The day of the month. The valid values for this member are 1 through 31.

### wHour

The hour. The valid values for this member are 0 through 23.

### wMinute

The minute. The valid values for this member are 0 through 59.

### wSecond

The second. The valid values for this member are 0 through 59.

### wMilliseconds

The millisecond. The valid values for this member are 0 through 999.

---

## sx\_color

It is base structure to define colors in different color spaces and formats. Base structure can be used to set

color property of object in “none” value. If you need define color within grayscale, RGB or CMYK color space, you need to use derived structures:

- **sx\_color\_i\_g** – grayscale color space, component values are integer in range 0..255
- **sx\_color\_i\_rgb** – RGB color space, component values are integer in range 0..255
- **sx\_color\_i\_rgba** – RGBA color space, component values are integer in range 0..255
- **sx\_color\_i\_cmyk** – CMYK color space, component values are integer in range 0..255
- **sx\_color\_f\_g** – grayscale color space, component values are float in range 0.0-1.0
- **sx\_color\_f\_rgb** – RGB color space, component values are float in range 0.0-1.0
- **sx\_color\_f\_rgba** – RGBA color space, component values are float in range 0.0-1.0
- **sx\_color\_f\_cmyk** – CMYK color space, component values are float in range 0.0-1.0

### Type Property

The property retrieves color space and format of color components. It is readonly property.

## SX::Graphics Namespace

### IRegion Interface

The interface describes an area (region) at the page. Region is a collection of [quads](#). Regions are usually used in selection and hit-testing operations.

**Windows:** This interface is derived from [IRefObject](#) at this platform.

### IBitmapData Data

The interface represents a raster image (bitmap). You can load this object from file or stream (see `IAppFactory::CreateBitmapFromStream` method) or you can get bitmap object from some methods of SDK interfaces, for example `IPage::CopyToBitmap`.

**Windows:** This interface is derived from [IRefObject](#) at this platform.

## Width Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves bitmap width, in pixels.

## Height Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves bitmap height, in pixels.

## DPI Property

Type: [sx\\_double](#). Access: full.

The property retrieves DPI of bitmap. When DPI is undefined, the value can be 0.

## PixelFormat Property

Type:PixelFormat. Access: readonly.

The property retrieves pixel format of bitmap (color space and bit depth). Use CopyAs method to convert bitmap in specified format.

## Palette Property

Type: IPaletteData. Access: readonly.

The property retrieves palette data for indexed formats of bitmap (see PixelFormat property).

## GetPixels Method

The method retrieves the address of bitmap pixel data in the memory. The stride of lines padded out to a byte. The method can return null if data is too big for placing in memory, use GetScanLine method in this case.

**DotNET:** The method retrieves address of unmanaged memory on this platform.

**UWP:** The method isn't available on this platform.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## GetScanLine Method

The method retrieves the address of pixel data for specified scan line of bitmap.

**DotNET:** The method retrieves address of unmanaged memory on this platform.

### Line Parameter

Type: [sx\\_uint](#).

The index of a scan line. Can be greater than or equal to 0 and less than bitmap height.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## CopyAs Method

The method creates new bitmap object and converts bitmap pixel data to specified format if it necessary. It returns IBitmapData interface.

### Format Parameter

Type:PixelFormat.

The preferred pixel format of new bitmap.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## SaveToStream Method

The method saves bitmap object to specified stream.

### Stream Parameter

Type: [ISequentialStream](#).

The stream for save. Please reset stream before save.

### Format Parameter

Type: ImageFormat.

The preferred file format of saved image (PNG, JPEG, etc.).

### Quality Parameter

Type: sx\_uint.

The compression level when you save a JPEG image. Can be greater than 0 (bad quality / good



compression) and less than or equal to 100 (good quality / bad compression).

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

# SX::Viewer Namespace

**Viewer** namespace is sub-namespace of [main SDK namespace](#), it contains declarations of enumerations, structures and interfaces used in the [document viewer](#).

## IWinViewer Interface

This interface is available for applications written in C++ as the classic Windows x86- and/or x64 programs. It represents a normal Windows window which contains a viewer object (represented as [Viewer interface](#)). You can create new instance of viewer window using [IAppFactory::CreateWinViewer](#) method. The viewer window you can place in the window hierarchy of your application.

## Viewer Property

Type: [IViewer](#). Access: readonly.

The multifunctional viewer object.

## Window Property

Type: `sx_window` (typedef of `HWND`). Access: readonly.

The window handle for a viewer window.

## Clipboard Property

Type: `ISystemClipboard`. Access: readonly.

The property provides an interface for interaction with system clipboard.

## EnableBarScroll Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of scrollbar events and scrolling of document in the viewer.

## EnableKeysScroll Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of keyboard events and scrolling of document in the viewer.

## EnableWheelScroll Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of mouse wheel events and scrolling of document in the viewer.

## SetEventHandler Method

The method sets a callback handler of viewer events.

### EventHandler Parameter

Type: [IViewerEvents](#).

The callback interface of event handler implemented by application.

## EnableZoomGesture Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of touchscreen (if present) events for zooming of viewer content.

## EnableScrollGesture Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of touchscreen (if present) events for scrolling of viewer content.

## EnableRotateGesture Property

Type: [sx\\_bool](#). Access: full.

The property enables processing of touchscreen (if present) events for the page rotation in viewer.

## IViewer Interface

---

**Windows:** the interface used on this platform only. It represents main properties and functionality of viewer. We highly recommend you [create the window based viewer](#) where you can get IViewer interface using Viewer property of [IWinViewer](#) interface. As an alternative for professionals, you can create viewer object without Windows window (using [IAppFactory::CreateViewer](#) method) and use this object for embedding in your own window.

## Document Property

Type: [IViewableDocument](#). Access: full.

The property specifies a document displayed in the viewer. Please note that you may not attach one IViewableDocument object in two or more viewers simultaneously. If you want display one document in the several viewers, you must request the necessary number of IViewableDocument objects (one for every viewer).

## Navigate Method

The method is an alternative for Document property. Using this method you can load new document and navigate to a specified page.

### Doc Parameter

Type: [IViewableDocument](#).

The document to be loaded.

### Page Parameter

Type: [sx\\_uint](#).

The index of page in the document. The viewer navigates to this page immediately.

### TakeSnapshot Parameter

Type: [sx\\_bool](#).

If this parameter is true, the viewer makes "snapshot" of viewer state (displayed document, layout and scroll properties of viewer) and places it in the viewer history. Using history functionality you can go back to this document.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Navigate Method

The method allows to restore previously fixed state of viewer (displayed document, layout and scroll properties of viewer) from screenshot object, see **TakeSnapshot** method also.

## Snapshot Parameter

Type: `IViewerSnapshot`.

The snapshot object to be loaded.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## TakeSnapshot Method

The method makes “snapshot” of viewer state (displayed document, layout and scroll properties of viewer). It returns the `IViewerSnapshot` object.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## Width Property

Type: [sx\\_uint](#). Access: readonly.

The property provides viewport width in [viewport units](#).

---

## Height Property

Type: [sx\\_uint](#). Access: readonly.

The property provides viewport height, in [viewport units](#).

---

## Resize Method

The method changes viewport size of the viewer.

Please **don't call this method directly** if you use a [window based viewer](#).

## Width Parameter

Type: [sx\\_uint](#).

The new viewport width.

## Height Parameter

Type: [sx\\_uint](#).

The new viewport height.

---

## Redraw Method

The method redraws actual viewport image on the specified drawing context. It returns false if viewport was completely drawn or true if drawing isn't complete and you must call this method until its completed.

Please **don't call this method directly** if you use a [window based viewer](#).

### Ctx Parameter

Type: Graphics::IDrawingContext.

The target drawing context.

### FirstBlock Parameter

Type: [sx\\_bool](#).

True if you want begin to draw viewport image or false if you want continue uncomplete drawing (see above description of return value).

### DrawRect Parameter

Type: [sx\\_rect](#).

This optional parameter is for internal use. Must be a null.

---

## Relayout Method

The method recalculates layout of displayed document in the viewer if the document structure (number of pages, page size or page order) was changed.

### NewActPage Parameter

Type: [sx\\_uint](#).

The index of active page (see Canvas property above) after changing of document structure.

For example, before changes the active page was 7, than page 4 was deleted in document, after this change, you should call Relayout method and specify firsts parameter as 6. In this case, the viewer recalculates the document layout but preserves (if possible) the active page and scrolling position.

If active page no more available, set this parameter to 0.

### AdoptPageOffset Parameter

Type: [sx\\_bool](#).

True if you want preserve visible part of active page after relayout.

## Invalidate Method

The method invalidates the viewport image.

### Pages Parameter

Type: [sx\\_bool](#).

True if page order or page content was changed and page cache in the viewer must be cleared.

### PDFAnnots Parameter

Type: [sx\\_bool](#).

True if PDF annotations in the document was changed and viewport cache must be updated.

## Canvas Property

Type: [ViewerCanvas](#). Access: readonly.

The property provides the canvas of the viewer. Using this interface you can control layouting of the document in the viewer and navigate over the document.

## Config Property

Type: [ViewerConfig](#). Access: readonly.

The property provides the configuration settings of the viewer.

## ToolMode Property

Type: [tool\\_mode](#). Access: full.

The property represents actual tool mode of the viewer.

**Important!** Before you set the value of this property to something else than "tool\_mode\_none" you must define [handler of viewer events](#) and associate it with viewer.

## Focus Method

The method informs the viewer when window receives or loses the input focus.

Please **don't call this method directly** if you use a [window based viewer](#).

### Set Parameter

Type: [sx\\_bool](#).

True if window receives the focus.

---

## MouseActivity Method

The method informs the viewer about any mouse or touch events.

Please **don't call this method directly** if you use a [window based viewer](#).

### X Parameter

Type: [sx\\_uint](#).

The x-coordinate of the cursor. The coordinate is relative to the upper-left corner of the client area.

### Y Parameter

Type: [sx\\_uint](#).

The x-coordinate of the cursor. The coordinate is relative to the upper-left corner of the client area.

### Flags Parameter

Type: [sx\\_flags](#), see also mouse\_flags enumeration.

The type of mouse event and optional flags.

---

## KeyboardActivity Method

The method informs the viewer about any keyboard events.

Please **don't call this method directly** if you use a [window based viewer](#).

### K Parameter

Type: [sx\\_byte](#).

The virtual-key code of the key.

### Flags Parameter

Type: [sx\\_flags](#), see also key\_flags enumeration.

The type of keyboard event and optional flags.

---

## ShowPopup Method

The method shows or hides popup annotation within viewport. It is allowed if ToolMode property is tool\_mode\_none, tool\_mode\_tracker, tool\_mode\_select\_text or tool\_mode\_annot. The popup mode in viewer configuration must be enabled also.

### Page Parameter

Type: [sx\\_uint](#).

The index of page where popup annotation is placed.

## Popup Parameter

Type: PDF::IPopup.

The popup annotation object which must be shown or hidden.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## SetEventHandler Method

The method sets a callback handler of viewer events.

**Important!** Please use the same method from [IWinViewer](#) interface instead this method if you use a [window based viewer](#).

## EventHandler Parameter

Type: [IViewerEvents](#).

The callback interface of event handler implemented by application.

## IViewerCanvas

This interface represents viewed canvas - a document layouted in the viewer according to viewer settings. Also it provides basic viewer functions: navigation, scrolling, content marking, etc.

## ActivePage Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves the index of active page - it is partially or completely visible page.

## Columns Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves the actual number of columns in layout.

## Fit Property

Type: viewer\_fit. Access: full.

The property specifies how a viewer will automatically zoom the pages to fit the size of window.



---

## Height Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves height of of the canvas, in [viewport units](#).

---

## HorizontalOffset Property

Type: [sx\\_uint](#). Access: full.

The property specifies horizontal offset of the viewport in the canvas coordinate system, in [viewport units](#).

---

## LayoutMultiplicity Property

Type: [sx\\_uint](#). Access: full.

The property specifies how many page columns (vertical layout) or page rows (horizontal layout) must have the layouted canvas.

---

## MarkingType Property

Type: `marking_type`. Access: readonly.

The property retrieves the type of marked object. This property is used with **tool\_mode\_none** and **tool\_mode\_select\_text** modes only.

---

## MarkingRegion Property

Type: [Graphics::IRegion](#). Access: readonly.

The property retrieves the region where marked object is located. The coordinates are in [coordinate system of page in the viewer](#).

---

## MarkedPage Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves the index of the page marked object is located.

---

## MarkingText Property

Type: [IStr](#). Access: readonly.

The property retrieves the marked text if MarkingType is **marking\_type\_text**.

---

## Rotation Property

Type: `sx_rotation_angle`. Access: full.

The property specifies the orientation (rotation angle) of the pages.

## Rows Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves the actual number of rows in layout.

## Thumbs Property

Type: `IViewerThumbs`. Access: readonly.

The property retrieves extension of canvas interface for thumbnail mode of viewer (**viewer\_mode\_thumbs** value of [ViewerConfig::Mode](#)).

## SeparateCover Property

Type: [sx\\_bool](#). Access: full.

The property specifies that first page must be single in the first row of multicolumn layout.

## VerticalLayout Property

Type: [sx\\_bool](#). Access: full.

The property specifies the vertical or horizontal direction of layouting.

## VerticalOffset Property

Type: [sx\\_uint](#). Access: full.

The property specifies vertical offset of the viewport in the canvas coordinate system, in [viewport units](#).

## VisiblePages Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves the number visible (partially or completely) pages. See also the **GetVisiblePage** method.

## Width Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves width of the canvas, in [viewport units](#).

## Zoom Property

Type: [sx\\_double](#). Access: full.

The property specifies the magnification scale of the pages in the viewer.

## CanvasToPage Method

The method transform coordinates from [canvas coordinate system](#) to [coordinate system of viewed page](#). For transformation in the reverse order, use **PageToCanvas** method. See also **GetPageMatrix** method.

### Index Parameter

Type: [sx\\_uint](#).

The index of a page, you can use **GetPageIndex** to get it.

### PtCanvas Parameter

Type: [sx\\_point](#).

The coordinates on the canvas (input parameter).

### PtPage Parameter

Type: [sx\\_point](#).

The coordinates on the specified page (output parameter).

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## ClearHistory Method

The method clears history of the viewer operations (scrolling, navigation, layout, etc). See also **History** method.

## GetPage Method

The method retrieves a page index for the specified column and row indexes.

### Row Parameter

Type: [sx\\_uint](#).

The row index in the page layout. It is zero-based.

### Col Parameter

Type: [sx\\_uint](#).

The column index in the page layout. It is zero-based.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## GetPageColumn Method

The method retrieves a column index of the page in the layout.

### Index Parameter

Type: [sx\\_uint](#).

The page index.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## GetPageIndex Method

The method retrieves a page located by specified coordinates on the canvas. It returns a page index or -1, if no page there.

### X Parameter

Type: [sx\\_uint](#).

The x-coordinate in the [canvas coordinate system](#).

### Y Parameter

Type: [sx\\_uint](#).

The y-coordinate in the [canvas coordinate system](#).

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## GetPageMatrix Method

The method retrieves a matrix for transformation of coordinates from [coordinate system of viewed page](#) to [canvas coordinate system](#). Method returns false if the viewer is in Flip mode and page located out of flip view canvas. See also **PageToCanvas** and **CanvasToPage** methods, its use the same matrix for transformation.

### Index Parameter

Type: [sx\\_uint](#).

The index of a page.

### M Parameter

Type: [sx\\_matrix](#).

The matrix for transformation (output parameter).

---

## GetPageRect Method

The method provides rectangle area of a specified page on the canvas. Method returns false if the viewer is in Flip mode and page located out of flip view canvas.

### Index Parameter

Type: [sx\\_uint](#).

The index of a page.

### Rect Parameter

Type: `sx_rect`.

The page rectangle (output parameter), coordinates are in [canvas coordinate system](#).

---

## GetPageRow Method

The method retrieves a row index of the page in the layout.

### Index Parameter

Type: [sx\\_uint](#).

The page index.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## GetVisiblePages Method

The method retrieves the index of specified visible page. It returns the index of page in the document.

### Index Parameter

Type: [sx\\_uint](#).

The index of page in the list of visible pages. Must be smaller then **VisiblePages** property.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## History Method

The method checks history list or navigates in the history of viewer. See also **ClearHistory** method.

## Back Parameter

Type: [sx\\_bool](#).

The backward direction in the history.

## Go Parameter

Type: [sx\\_bool](#).

If this parameter is false, method returns availability of steps in the history in specified direction. If parameter is true and at least one step is available, the viewer navigates one step in specified direction.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## Navigate Method

The method navigates to the specified location on the specified page.

## Index Parameter

Type: [sx\\_uint](#).

The page index.

## X Parameter

Type: [sx\\_uint](#).

The x-coordinate on the page (in [viewer page coordinate system](#)).

## Y Parameter

Type: [sx\\_uint](#).

The y-coordinate on the page (in [viewer page coordinate system](#)).

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## Navigate Method

The method navigates to the specified rectangle area on the specified page.

## Index Parameter

Type: [sx\\_uint](#).

The page index.

## Rect Parameter

Type: [sx\\_rect](#).

The rectangle area on the page (coordinates are in [viewer page coordinate system](#)).

## EnsureVisible Parameter

Type: [sx\\_bool](#).

If parameter is false, method navigates always, if parameter is true and rectangle area is visible now the method makes nothing.

## ZoomByRect Parameter

Type: [sx\\_bool](#).

If parameter is true the viewer changes zoom to fit rectagle area.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## PageToCanvas Method

The method transform coordinates from [coordinate system of viewed page](#) to [canvas coordinate system](#). For transformation in the reverse order, use **CanvasToPage** method. See also **GetPageMatrix** method.

## Index Parameter

Type: [sx\\_uint](#).

The index of a page.

## PtPage Parameter

Type: [sx\\_point](#).

The coordinates on the specified page (input parameter).

## PtCanvas Parameter

Type: [sx\\_point](#).

The coordinates on the canvas (output parameter).

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## ResetMarking Method

The method sets ot cleares marking in the viewer.

## Marking Parameter

Type: [Graphics::IRegion](#).

The region of marking. Can be null.

## Index Parameter

Type: [sx\\_uint](#).

The index of a page where the marking located.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## SnapZoom Method

The method changes zoom discretely. It returns true if zoom was changed.

## Index Parameter

Type: [sx\\_bool](#).

Decrease or increase the zoom.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## IViewerConfig

This interface provides viewer configuration options and settings.

---

## IViewableDocument

The interface represents viewable document. For the documents implemented in the PDF Xpansion SDK, you can get this interface using GetViewableDocument method of [IBaseDocument interface](#). Please note that you may not attach one instance of this interface in two or more viewers simultaneously. If you want display one document in the several viewers, you must request the necessary number of instances (one for every viewer).



**Windows:** This interface is derived from [IRefObject](#) at this platform. We recommend you use an instance of an [ObjPtr template class](#) where you can use an IBaseDocument pointer in a safe manner.

**Important!**

- You may not use this interface for drawing of pages directly in your application, please use appropriate methods of native page interfaces.
- You may implement this interface for your own document format, than you can load you documents in the [viewer](#). The actual viewer uses Direct2D technology for rendering of content, see DrawPage method for detailed information.

## GetUPI Method

The method retrieves “units per inch” property of the document. Is a measure of units in the document coordinate system, in particular the number of units that can be placed within the span of 1 inch.

## GetPageCount Method

The method retrieves number of pages in the document.

## GetPageSize Method

The method retrieves size of the requested page, the size is in document units.

## DrawPage Method

**Windows:** The method declared on this platform only, you must implement rendering of specified page with specified transformation to the provided graphic context.

The actual viewer implementation uses Direct2D technology for rendering of content, therefore this method must draw content to the ID2D1RenderTarget context.

## IViewerEvents Interface / Delegate

Using power and effective event mechanism provided by the viewer, your application can extend standard functionality of viewer and perform the appropriate application logic for visual processing of PDF or other documents.

Many useful tools of the viewer (see **ToolMode** viewer property) can be used together with event mechanism only, because the application must control these tools using own event handler. Please separate viewer tool events using [IViewerEvent::Type](#) property value **viewer\_event\_tool\_activity**.

**Windows:** At this platform you must implement `IViewerEvents` interface in your application, if you want to process different viewer events and perform your tasks. Call **`SetEventHandler`** method of [IWinViewer](#) or [IViewer](#) interfaces to specify an instance of your `IViewerEvents` implementation as an event handler.

**DotNET:** At this platform the `IViewerEvents` delegate is declared instead of callback interface. Please define an event handler method with the same signature as the `IViewerEvents` delegate and add an instance to the **`OnEvent`** event of [Forms](#) or [WPF](#) control.

**UWP:** At these platforms the `IViewerEvents` delegate is declared instead of callback interface. Please define an event handler method with the same signature as the `IViewerEvents` delegate and add an instance to the **`OnEvent`** event of [WinStore](#) or [WinPhone](#) control.

## OnEvent Method / Event

The viewer calls this application defined handler to inform it about different events occurred in the document viewer or produced by end user activity. The handler must return false value.

### Event Parameter

Type: [IViewerEvent](#).

The event specific data.

## IViewerEvent Interface

This interface describes event raised by document viewer and provides event specific data.

### Type Property

Type: [viewer\\_event](#). Access: readonly.

The property provides the type of event. Using this property you can type cast the **`IViewerEvent`** interface to the derived interfaces:

- [IViewerEvent\\_PDF](#) (for `viewer_event_pdf`)
- [IViewerEvent\\_XPS\\_Link](#) (for `viewer_event_xps_link`)
- [IViewerEvent\\_PopupActivity](#) (for `viewer_event_popup`)
- [IViewerEvent\\_SetActivePage](#) (for `viewer_event_set_page`)

- [IViewerEvent\\_ScrollCanvas](#) (for viewer\_event\_scroll\_canvas)
- [IViewerEvent\\_MouseActivity](#) (for viewer\_event\_mouse\_activity)
- [IViewerEvent\\_KeyboardActivity](#) (for viewer\_event\_keyboard\_activity)
- [IViewerEvent\\_ToolActivity](#) (for viewer\_event\_tool\_activity)

## Handled Property

Type: [sx\\_bool](#).

The property indicating whether the event was handled. True to bypass the viewer's default handling; otherwise, false to also pass the event along to the default viewer handler.

## IViewerEvent\_PDF Interface

This interface provides the events described by PDF ISO standard and raised by document viewer. See also derived interface [IViewerEvent\\_PDF\\_Annot](#) for the PDF annotation events.

## PDFType Property

Type: PDF::pdf\_event. Access: readonly.

The property retrieves type of PDF event.

## IViewerEvent\_PDF\_Annot Interface

This interface provides the annotation events described by PDF ISO standard and raised by document viewer. It is derived from [IViewerEvent\\_PDF](#).

## Annot Property

Type: [PDF::IAnnot](#). Access: readonly.

The property retrieves PDF annotation object.

## IViewerEvent\_XPS\_Link Interface

This interface provides the navigation event for XPS documents.

## Activity Property

Type: hotspot\_flags. Access: readonly.

The property retrieves the type of end-user activity relative to hyperlink area (pointer status: enter / down / up / exit).

## Link Property

Type: XPS::ILink. Access: readonly.

The property retrieves hyperlink object.

## IViewerEvent\_PopupActivity Interface

This interface provides the popup events in the viewer.

## Activity Property

Type: popup\_activity. Access: readonly.

The property retrieves the operation with popup:

- popup\_activity\_visibility - show / hide popup
- popup\_activity\_activate - popup gets input focus
- popup\_activity\_rect - popup area on the page was changed
- popup\_activity\_text - popup text was changed

## Visible Property

Type: [sx\\_bool](#). Access: readonly.

The visibilty status of popup.

## Active Property

Type: [sx\\_bool](#). Access: readonly.

The popup has input focus now.

## Popup Property

Type: PDF::IPopup. Access: readonly.

The reference to PDF popup annotation. Can be null (for non-PDF popups).

## GetViewportRect Method

The method retrieves the popup area relative to the viewport coordinate system.

## Rect Parameter

Type: [sx\\_rect](#).

The popup rectangle area.

## IViewerCancelableEvent Interface

The interface provides base functionality for all cancelable event types. It is derived from [IViewerEvent](#).

### Canceled Property

Type: [sx\\_bool](#). Access: readonly.

The property value indicating whether the event should be canceled.

### Cancel Method

The method declares that the event should be canceled.

## IViewerEvent\_SetActivePage

The interface provides event which informs about intention to change active page. It is derived from [IViewerCancelableEvent](#).

Because this is cancelable event, you can restrict changing of active page.

### Page Property

Type: [sx\\_uint](#). Access: readonly.

The property value retrieves an index of new page.

## IViewerEvent\_ScrollCanvas Interface

The interface provides event which informs about intention to scroll a document canvas relative to the viewport. It is derived from [IViewerCancelableEvent](#).

Because this is cancelable event, you can restrict scrolling.

### VerticalOffset Property

Type: [sx\\_uint](#). Access: readonly.

The property value retrieves new vertical offset.

### HorizontalOffset Property

Type: [sx\\_uint](#). Access: readonly.

The property value retrieves new horizontal offset.

## IViewerEvent\_MouseActivity Interface

The interface provides the mouse, touch, stylus input interactions. It is derived from [IViewerCancelableEvent](#). Because this is cancelable event, you can skip an input action.

### Activity Property

Type: `mouse_flags`. Access: readonly.

The property retrieves the type of input interaction

### X Property

Type: `sx_uint`. Access: readonly.

The property value retrieves x-coordinate of pointer in the viewport coordinate system.

### Y Property

Type: `sx_uint`. Access: readonly.

The property value retrieves y-coordinate of pointer in the viewport coordinate system.

### Delta Property

Type: `sx_int`. Access: readonly.

The property value that retrieves the amount that the mouse wheel has changed.

### Ctrl Property

Type: `sx_bool`. Access: readonly.

The property that indicates the CTRL key is down.

### Shft Property

Type: `sx_bool`. Access: readonly.

The property that indicates the SHIFT key is down.

## IViewerEvent\_KeyboardActivity Interface

The interface provides the keyboard input. It is derived from [IViewerCancelableEvent](#). Because this is cancelable event, you can skip an input action.

---

## Key Property

Type: [sx\\_byte](#). Access: readonly.

The property value retrieves the key code.

**Important!** The key codes are mapped according to the layout of specified platform (see [ViewerConfig::KeyboardLayout](#) property).

---

## Pressed Property

Type: [sx\\_bool](#). Access: readonly.

The property that indicates the specified key is pressed (or released) now.

---

## RepeatCount Property

Type: [sx\\_uint](#). Access: readonly.

The property value is the number of times the keystroke is autorepeated as a result of the user holding down the key.

---

## Shft Property

Type: [sx\\_bool](#). Access: readonly.

The property that indicates the SHIFT key is pressed now.

---

## Ctrl Property

Type: [sx\\_bool](#). Access: readonly.

The property that indicates the CTRL key is pressed now.

---

## Alt Property

Type: [sx\\_bool](#). Access: readonly.

The property that indicates the ALT key is pressed now.

---

## RightCtrl Property

Type: [sx\\_bool](#). Access: readonly.

The property that indicates the right CTRL key is pressed now.

---

## RightAlt Property

Type: [sx\\_bool](#). Access: readonly.

The property that indicates the right ALT key is pressed now.

## IViewerEvent\_ToolActivity Interface

The interface provides the events of the viewer tools. You must implement processing of this event because the viewer tools can't work without handling of its events. The interface is derived from [IViewerCancelableEvent](#). Because this is cancelable event, you can skip many tool actions according to your criteria.

### State Property

Type: [IToolState](#). Access: readonly.

The property retrieves the interface of active tool.

**Important!** You may cache this interface (active tool) and use it between the tool events, but not later than the tool mode (see ToolMode property of the viewer) will be changed.

## IToolState Interface

The interface represents active viewer tool and actual tool state.

**Important!** This interface represents a tool object within the viewer until the other tool will be activated or other document will be loaded to the viewer. Therefore you may cache this interface and use it between the tool events, but not later than the tool mode will be changed (see **sx\_tool\_change** description below).

### ToolMode Property

Type: [tool\\_mode](#). Access: readonly.

The property retrieves the type of actual tool, it duplicates the similar property of viewer. Using this property you can type cast the **IToolState** interface to the derived interfaces:

- IToolTracker (for tool\_mode\_tracker)
- IToolRectMark (for tool\_mode\_rect\_mark)
- IToolSnapshot (for tool\_mode\_snapshot)
- IToolLens (for tool\_mode\_lens)
- IToolTextSelect (for tool\_mode\_select\_text)
- IToolStylus (for tool\_mode\_stylus)
- IToolAnnot (for tool\_mode\_annot)
- IToolFreeText (for tool\_mode\_free\_text)



- IToolInk (for tool\_mode\_ink)
- IToolEraser (for tool\_mode\_eraser)
- IToolShape (for tool\_mode\_shape)

## Activity Property

Type: `sx_tool_act`. Access: readonly.

The property provides the status of tool activity.

- **sx\_tool\_change** - this status notifies about activating new tool in the viewer, under this event you can setup new tool and/or cache tool interface in your application. If tool type is "tool\_mode\_none", than you may cache tool interface because in is temporary object.
- **sx\_tool\_start** - the tool workflow is started
- **sx\_tool\_apply** - the tool workflow is successfully finished
- **sx\_tool\_cancel** - the tool workflow is canceled

## Page Property

Type: [sx\\_uint](#). Access: readonly.

The property value retrieves an index of page where tool is loaded now. It can be "-1" if tool workflow is not started yet.

## Reset Method

The method breaks the tool workflow if it started already.

## Enumerations

### Viewer\_flags Enumeration

This enumeration defines optional styles of the [viewer window](#).

#### viewer\_flags\_scrollbar\_always

The window scrollbars must be visible always.

#### viewer\_flags\_scrollbar\_autohide

The window has the scrollbars and hide its if no scroll is possible.

#### viewer\_flags\_border\_flat

The window has the flat border.

### **viewer\_flags\_border\_3D**

The window has the 3D style border.

---

## **Tool\_mode Enumeration**

This enumeration defines tool modes of the document viewer. Every tool mode implements own logics of reaction to end-user activities in the viewer. For example, you need "tool\_mode\_select\_text" if you want allow to select text on the pages interactively.

### **tool\_mode\_none**

The default mode of the viewer, it supports optionally the link navigation, PDF form filling and PDF defined actions. Please note, other tool modes don't support the link navigation, PDF form filling and PDF defined actions.

### **tool\_mode\_custom**

This mode provides the possibility to implement completely own logics for processing of end-user activity in the viewer.

### **tool\_mode\_tracker**

This mode provides the possibility to specify place or rectangle area on the page in the viewer.

This mode can be used for adding sticky notes, stamps, watermarks and similar PDF annotations.

### **tool\_mode\_rect\_mark**

This mode provides the possibility to mark rectangle zones on the pages of viewed document. The tool supports single or multiple zones on the page, adding, moving and resize of existing zones.

### **tool\_mode\_snapshot**

This mode provides the snapshot tool.

### **tool\_mode\_lens**

This mode provides the lens tool.

### **tool\_mode\_measure**

This mode provides the measure tool, it is on construction yet.

### **tool\_mode\_select\_text**

This mode provides the possibility to select text on single page in the viewer. This tool can be used for copying in the clipboard, marking keyword placement and text markup in PDF documents.

## **tool\_mode\_annot**

This mode provides the possibility to manage different types of PDF annotations. The tool supports single or multiple selection, moving and resize of annotations. Using this tool you can implement removing annotations, view or edit annotation properties (in own window or form).

## **tool\_mode\_free\_text**

This mode provides the possibility to create or edit FreeText type of PDF annotations (textbox and callout).

## **tool\_mode\_ink**

This mode provides the possibility to create or edit Ink type of PDF annotations.

## **tool\_mode\_eraser**

This mode provides the possibility to edit Ink type of PDF annotations.

## **tool\_mode\_shape**

This mode provides the possibility to create or edit shape types of PDF annotations (line, rectangle, circle, polyline, polygon).

---

## **Viewer\_event Enumeration**

This enumeration defines types of events used by the document viewer.

### **viewer\_event\_attach\_doc**

New document was loaded in the viewer.

### **viewer\_event\_detach\_doc**

The viewed document was unloaded from the viewer.

### **viewer\_event\_invalidate**

The viewport was invalidated and will be redrawn.

### **viewer\_event\_change\_act\_page**

The active page is changed.

### **viewer\_event\_change\_page\_visibility**

The visible page(s) was changed.

### **viewer\_event\_change\_layout**

The document was relayouted.

### **viewer\_event\_change\_scroll**

The document canvas was scrolled.

### **viewer\_event\_change\_config**

The viewer configuration was changed.

### **viewer\_event\_change\_marking**

The marking was changed or cleared.

### **viewer\_event\_change\_thumb\_marking**

The page(s) was selected or unselected.

### **viewer\_event\_pdf**

The PDF specific event.

### **viewer\_event\_xps\_link**

The XPS navigation event.

### **viewer\_event\_popup\_activity**

The event of the viewer popup.

### **viewer\_event\_set\_page**

The cancelable event - try to navigate.

### **viewer\_event\_scroll**

The cancelable event - try to scroll.

### **viewer\_event\_mouse\_activity**

The cancelable event - mouse / touch / stylus activity of end-user.

### **viewer\_event\_keyboard\_activity**

The cancelable event - keyboard input of end-user.

### **viewer\_event\_tool\_activity**

The cancelable event of the viewer tool.

---

## **SX::PDF Namespace**

### **Coordinate System and Unit of Measure**

PDF defines own coordinate system that is called user space. The positive x axis extends horizontally to the

right and the positive y axis vertically upward, as in standard mathematical practice. In most cases, the origin of the coordinate system (0,0) coincides with the lower left corner of the page, but this is not mandatory. As a rule, the unit of measure is the point (1/72 of an inch). But every page may have optional property - a multiplicative factor that shall give the size of page user space units, in multiples of point (1/72 of an inch), see [IPage::UserUnit](#). Please note, **y-member** of [sx\\_rect structure](#) contains lowest coordinate of specified area, it is bottom side of rectangle within PDF coordinate system.

## IDocument Interface

The interfaces provides interface of PDF document object implemented by SDK. This interface is derived from [IBaseDocument](#). You can create new instance of PDF document object using [IAppFactory::CreatePDFDocument](#) method.

**Windows:** We highly recommend you use an instance of an [ObjPtr template class](#) where you can use an [PDF::IDocument](#) pointer in a safe manner.

## FilePath Property

Type: [sx\\_str](#). Access: readonly.

The property provides file path of the document if it opened from file or saved to file, otherwise the property is null.

## Properties Property

Type: [IDocProperties](#). Access: readonly.

The property provides the document properties.

## Pages Property

Type: [IPages](#). Access: readonly.

The property provides the pages collection.

## Outline Property

Type: [IOutlineItem](#). Access: readonly.

The property provides root item of the document outline. It produces an exception [ErrorAuthFailed](#) if your license does not permit to use outline elements.

## NamedDests Property

Type: INamedDests. Access: readonly.

The property provides a collection of named destinations in the document. It produces an exception [ErrorAuthFailed](#) if your license does not permit to use named destinations.

## FormFields Property

Type: [IFormFields](#). Access: readonly.

The property provides an interactive form of the document. It produces an exception [ErrorAuthFailed](#) if your license does not permit to use form fields.

## Layers Property

Type: ILayers. Access: readonly.

The property provides a collection of layers in the document. It produces an exception [ErrorAuthFailed](#) if your license does not permit to use layers.

## Files Property

Type: IEmbeddedFiles. Access: readonly.

The property provides a collection of files embedded in the document. It produces an exception [ErrorAuthFailed](#) if your license does not permit to use embedded files.

## Resources Property

Type: IDocResources. Access: readonly.

The property provides the document resources.

## OpenStream Method

The method loads the PDF document from stream.

### Stream Parameter

Type: [ISequentialStream](#).

The document data stream. Please reset stream before load.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## OpenFile Method

The method loads the document from file.

### FilePath Parameter

Type: [sx\\_str](#).

Fully qualified path of the file to open. The file must have read access.

### MaxCache Parameter

Type: [sx\\_uint](#).

Max size of memory cache. If file size is less than this parameter, the file will be completely cached in memory and immediately closed after calling this method. In other case, the file is still open and locked until the document object will be destroyed or document saved.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## SelectRevision

The method selects specified revision (version) of document. We recommend you use this method immediately after calling OpenStream or OpenFile because method works until the document still in nonmodified state, otherwise it is failed. See also IDocProperties::RevisionsCount and IDocProperties::CurrentRevision properties.

### RevIndex Parameter

Type: [sx\\_uint](#).

Zero-based index of revision. Must be less then IDocProperties::RevisionsCount.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## SaveAsStream

The method saves the PDF document to the stream.

### Stream Parameter

Type: [ISequentialStream](#).

The stream for save. Please reset stream before save.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## SaveAsFile

The method saves the document to the file.

### FilePath Parameter

Type: [sx\\_str](#).

Fully qualified path of the file to save. Method creates a new file, always. If the specified file exists it must have write access.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## SaveRevisionAsStream

The method appends new revision to the PDF document (any changes or digital signature as an incremental update). It fails if document is non-modified or wasn't loaded from the stream or file.

**Please note:** the method writes data of complete document to the stream, not an incremental update data only.

### Stream Parameter

Type: [ISequentialStream](#).

The stream for save. Please reset stream before save.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## SaveRevisionAsFile

The method appends new revision to the PDF document (any changes or digital signature as an incremental update). It fails if document is non-modified or wasn't loaded from the stream or file.

**Please note:** the method writes data of complete document to the file, not an incremental update data only.

### FilePath Parameter

Type: [sx\\_str](#).

Fully qualified path of the file to save. Method creates a new file, always. If the specified file exists it must have write access.

### Errors



The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## StartCombine

The method checks the possibility to combine with other document (copy pages and other elements from specified document). It prepares combining operation also. You can use `CombinePage` or `CombineAllPages` methods to specify pages to be combined. You must call the `EndCombine` method to comply combine operation.

Please note! Using the combine methods you can not only combine different types of documents (PDF, XPS, etc.) but also convert PDF to XPS and XPS to PDF.

### SourceDocument Parameter

Type: [IBaseDocument](#).

The pages (and other elements as links or annotations) of `SourceDocument` will be copied in current document.

### CombineOptions Parameter

Type: `sx_flags`, see also [combine](#) enumeration.

The parameter specifies combine options.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## CombinePage

The method copies one page from `SourceDocument` (see `StartCombine` method above) to the current document, optionally with PDF annotations. The method can't be called twice for the same page. Also it cannot be called together with `CombineAllPages` method.

### From Parameter

Type: [sx\\_uint](#).

The page index in `SourceDocument`. Can be greater than or equal to 0 and less than page count.

### To Parameter

Type: [sx\\_uint](#).

The position in the current document where a copied page will be inserted. Can be greater than or equal to 0 and less than or equal page count.

### CombineOptions Parameter

Type: `sx_flags`, see also [annot\\_type](#) enumeration.

The parameter specifies the types of PDF annotations to be copied. You can combine the enumeration values by using the bitwise OR operator.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## CombineAllPages

The method copies all pages of SourceDocument (see StartCombine method above) to the current document, optionally with PDF annotations. It cannot be called together with CombinePage method.

## To Parameter

Type: [sx\\_uint](#).

The position in the current document where the copied pages will be inserted. Can be greater than or equal to 0 and less than or equal page count.

## CombineOptions Parameter

Type: `sx_flags`, see also [annot\\_type](#) enumeration.

The parameter specifies the types of PDF annotations to be copied. You can combine the enumeration values by using the bitwise OR operator.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## EndCombine

The method performs necessary actions to combine two documents after calling StartCombine and CombinePage/CombineAllPages methods.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## ConformToPDFA

The method converts the PDF document to make it conform to specified PDF/A format.

Please note that

➤ some correct PDF documents can't be correctly converted to PDF/A format and method fails, the

detailed reason you get using callback messaging (see information about third parameter below)

- any changes in the document after calling this method, can make document non conform to PDF/A, so we highly recommend you to save conform document immediately after successful calling of this method
- you must disable document encryption if available, before call this method

## Version Parameter

Type: pdfa.

The PDF/A version to which a document must be conform.

## Opt Parameter

Type: [sx\\_flags](#).

The reserved parameter, must be 0.

## Status Parameter

Type: [IStatusPDFa](#).

The callback interface, the parameter is optional, but strongly recommended. You need to implement it to get detailed information about converting procedure.

**UWP:** At these platforms the OnMessagePDFa event of Document class is declared instead of third parameter. Please define an event handler method and add an instance to this event.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

# IDocProperties Interface

The interface represents the properties of PDF document.

## ID1 Property

Type: [sx\\_str](#).

The property provides the permanent identifier of the document.

## ID2 Property

Type: [sx\\_str](#).

The property provides the changing identifier of the document.

## FormatVersion Property

Type: [sx\\_str](#).

The property provides the PDF format version, the valid version values are "1.0", "1.1", "1.2", "1.3", "1.4", "1.5", "1.6", "1.7", "2.0".

## Linearized Property

Type: [sx\\_bool](#). Access: readonly.

The property is true, if document was loaded from file or stream optimized for delivery from servers to web browsers over the Internet.

## Security Property

Type: [IDocSecurity](#). Access: readonly.

The property provides security properties of the document.

## IDocSecurity Interface

This interface manages the security properties of PDF document.

## Security Property

Type: doc\_security. Access: readonly.

The property determine whether security is enabled and type of PDF security. The values can be:

- doc\_security\_none - the document is not encrypted
- doc\_security\_undefined - the document is encrypted, encryption method is unknown
- doc\_security\_password - the document is encrypted, standard password based security
- doc\_security\_pki - the document is encrypted, standard public-key based security

## StandardSecurity Property

Type: [IStandardSecurity](#). Access: readonly.

The property provides the interface of two standard types of document security.

You may use this property if **Security** property equals doc\_security\_password (cast IStandardSecurity to

[IPasswordSecurity](#)) or it equals doc\_security\_pki (cast IStandardSecurity to IPKISecurity).

## HandlerName Property

Type: [sx\\_str](#).

The property provides the name of used security method.

## HandlerFormat Property

Type: [sx\\_str](#).

The property provides the format description of used security method.

## ChangeSecurity Method

The method sets or removes the document security.

### Sec Parameter

Type: doc\_security.

The new security type. You can't set doc\_security\_undefined value here.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## IStandardSecurity Interface

The interface represents the base properties of two standard types of document security. So two interfaces: [IPasswordSecurity](#) and IPKISecurity inherit it.

## FullControl Property

Type: [sx\\_bool](#). Access: readonly.

If property is true, unlimited access to the document is granted. This unlimited access includes the ability to change the document's security and access permission. In case of password based security, you must load document using owner password.

**Important!** The IDocSecurity::ChangeSecurity method allows you completely reset security without this permission.

## EncryptionAlgorithm Property

Type: encryption\_algorithm.

The property determines encryption algorithm. The values can be:

- encryption\_algorithm\_rc4 - RC4 encryption algorithm
- encryption\_algorithm\_aes - AES encryption algorithm

---

## EncryptionKeyLength Property

Type: encryption\_key\_length.

The property determines length of key used for encryption. The values can be:

- encryption\_key\_length\_40
- encryption\_key\_length\_128
- encryption\_key\_length\_256
- encryption\_key\_length\_384
- encryption\_key\_length\_512

---

## Permissions Property

Type: doc\_permissions.

The property provides the document permissions. The permissions are:

- doc\_permissions\_full\_control
- doc\_permissions\_normal\_print
- doc\_permissions\_modify\_content
- doc\_permissions\_copy\_content
- doc\_permissions\_modify\_annots
- doc\_permissions\_fill\_form
- doc\_permissions\_accessibility
- doc\_permissions\_assemble
- doc\_permissions\_quality\_print

The multiple permissions can be combined by using the bitwise OR operator.

---

## IPasswordSecurity Interface

The interface inherits [IStandardSecurity](#) interface in the case of use of doc\_security\_password security.

## SetOwnerPassword Method

The method sets owner password for encryption or decryption of secured PDF document. It returns boolean status of call.

You must call this method successfully for opening of encrypted document and getting full control over the document, otherwise it is enough to call SetUserPassword method successfully.

You must call this method successfully before save encrypted document.

### Password Parameter

Type: [sx\\_str](#).

The property specifies the owner password.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## SetUserPassword Method

The method sets user password for encryption or decryption of secured PDF document. It returns boolean status of call.

You need call this method successfully for opening of encrypted document and getting access to document according to [security permissions](#), if you need full acces to document you must call SetOwnerPassword method successfully.

You must call this method successfully before save encrypted document.

### Password Parameter

Type: [sx\\_str](#).

The property specifies the owner password.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## IPages Interface

The interface represents a pages collection of PDF document.

### Count Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves number of pages in the document.

## Item Method

The method retrieves the specified page. It returns [IPage](#) interface.

### Index Parameter

Type: [sx\\_uint](#).

The index of a page. Can be greater than or equal to 0 and less than page count.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Add Method

The method adds new empty page to the end of document. It returns [IPage](#) interface. **Note:** you must set [media box](#) of a page after adding.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Insert Method

The method inserts new empty page to the specified position. It returns [IPage](#) interface. **Note:** you must set [media box](#) of a page after inserting.

### To Parameter

Type: [sx\\_uint](#).

The position to insert a page. Can be greater than or equal to 0 and less than or equal to page count.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Copy

The method duplicates a page in this document or copies a page from other document (PDF or XPS) and inserts copied page to the specified position. It returns [IPage](#) interface.

### Page Parameter

Type: [IPage](#) or XPS::IPage.

The page to be copied.



## To Parameter

Type: [sx\\_uint](#).

The position to insert a page. Can be greater than or equal to 0 and less than or equal to page count.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## Move

The method moves a page to the specified position.

## From Parameter

Type: [sx\\_uint](#).

The index of a page to move. Can be greater than or equal to 0 and less than page count.

## To Parameter

Type: [sx\\_uint](#).

The position to insert a page. Can be greater than or equal to 0 and less than or equal to page count.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## Remove

The method removes a specified page from the document.

## Index Parameter

Type: [sx\\_uint](#).

The index of a page. Can be greater than or equal to 0 and less than page count.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## IndexOf

The method searches for the specified page object and returns the zero-based index of a page within the entire pages collection.

## Page Parameter

Type: [IPage](#).

The page object.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Compact

The method compacts memory used by page content and resources.

## CompactObjects Parameter

Type: `sx_flags`, see also [PDF::compact\\_objects](#).

The options of compact operation.

## IPage Interface

The interface represents a page of PDF document.

## Width Property

Type: [sx\\_uint](#). Access: readonly.

The width of a page, in [points](#). This property don't considers values of UserUnit and Rotation properties. See also [GetMediaBox](#) method. Use SetMediaBox or SetCropBox methods to change page width.

## Height Property

Type: [sx\\_uint](#). Access: readonly.

The height of a page, in [points](#). This property don't considers values of UserUnit and Rotation properties. See also [GetMediaBox](#) method. Use SetMediaBox or SetCropBox methods to change page height.

## Rotation Property

Type: `sx_rotation`.

The number of degrees (multiple of 90) by which the page shall be rotated clockwise when displayed or printed.

## UserUnit Property

Type: [sx\\_double](#).

A positive number that shall give the size of user space units, in multiples of point.

---

## Annots Property

Type: [IAnnots](#). Access: readonly.

The property provides the annotations collection of the page.

---

## Label Property

Type: [sx\\_str](#). Access: readonly.

The property provides the page label. See [IPages](#) methods to change labeling of the pages.

---

## Thumbnail Property

Type: [IRasterImage](#). Access: readonly.

The property provides the page thumbnail.

---

## GetMediaBox Method

The method retrieves a rectangle in [page user space](#), that shall define the boundaries of the page. The visible region of the page can be clipped, see [GetCropBox](#) method below.

### Box Parameter (or return parameter)

Type: [sx\\_rect](#).

The boundaries of the page.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## SetMediaBox Method

The method sets a rectangle in [page user space](#), that shall define the boundaries of the page. If the page has crop box and it is greater than new media box, the library inflates media box up to crop box, see [GetCropBox](#) method below.

### Box Parameter

Type: [sx\\_rect](#).

The boundaries of the page.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## GetCropBox Method

The method retrieves a rectangle in [page user space](#), that shall define the visible region of the page. If crop box isn't defined, the method retrieves media box of the page, see GetMediaBox method above.

### Box Parameter (or return parameter)

Type: [sx\\_rect](#).

The visible region of the page.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## SetCropBox Method

The method sets a rectangle in [page user space](#), that shall define the visible region of the page. If the new crop box is greater than media box of the page, the library deflates crop box up to media box, see GetMediaBox method above.

### Box Parameter

Type: [sx\\_rect](#).

The visible region of the page.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## SetThumbnail Method

The method renders new thumbnail of a page and saves it inside the document. Call the method with zero parameters to remove existing thumbnail. The method fails if you try to render a big thumbnail (with DPI > 18) without "Draw Page" permission in your license.

### MaxW / MaxH Parameters

The parameters specify the max size of new thumbnail in pixels.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## CalcMatrix Method

The method calculates a transformation matrix using properties of the page and specified parameters of external coordinate system (for example, unified page coordinate system of viewer). Using this matrix you can transform any coordinates from the page space to the external coordinate system and vice versa (using

inverted matrix).

## M Parameter

Type: [sx\\_matrix](#).

The transformation matrix, it is output parameter.

## Zoom Parameter

Type: [sx\\_double](#).

The scale value, where 1.0 is original size.

## Angle Parameter

Type: [sx\\_rotation](#).

The number of degrees (multiple of 90) by which the page shall be rotated.

## Dx Parameter

Type: [sx\\_double](#).

The horizontal position of the page in external coordinate system.

## Dy Parameter

Type: [sx\\_double](#).

The vertical position of the page in external coordinate system.

## DPI Parameter

Type: [sx\\_uint](#).

The DPI property external coordinate system.

---

## GetRichContent Method

The method provides direct access to the page content as a hierarchical structure of content objects. It retrieves [IRichContent interface](#).

## Reserved Parameter

The parameter reserved for future use.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

# IAnnots Interface

The interface represents an annotations collection of the page. Using this interface you can enumerate annotations, create or remove its, change order (Z-order on the page) of the annotations.

## Count Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves number of annotations on the document.

## Item Method

The method retrieves the specified annotation. It returns [IAnnot](#) interface.

## Index Parameter

Type: [sx\\_uint](#).

The index of an annotation. Can be greater than or equal to 0 and less than page count.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## FindItem Method

Method retrieves an annotation located at the coordinates of x and y. Most often used with pointer operations in the viewer to determine an annotation under the pointer. It returns [IAnnot](#) interface.

**Important!** The method enumerates annotations from last to first (according to visibility of overlapped annotations).

## P Parameter

Type: [sx\\_point](#).

The coordinates to find. They must be in [page user space](#).

## Mask Parameter

Type: [sx\\_flags](#), see also [annot\\_type](#) enumeration.

The type(s) of annotations to be found. You can combine the enumeration values by using the bitwise OR operator.

## FromAnnot Parameter

Type: [IAnnot](#).

The annotation returned by a previous call to this method, this parameter allow you to find all overlapped annotations. For first call it must be null.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## InsertNew Method

The method creates the annotation of specified type. It returns [IAnnot](#) interface. You must set necessary properties of annotation, at least the rectangle area.

### Type Parameter

Type: [sx\\_flags](#), see also [annot\\_type](#) enumeration.

The type of new annotation.

### Before Parameter

Type: [sx\\_uint](#).

The new annotation will be inserted into the collection before this annotation. Can be greater than or equal to 0 and equal to annotation count.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## InsertNewWidget Method

The method creates the widget annotation. It returns [IWidget](#) interface. You must set necessary properties of annotation, at least the rectangle area.

### Field Parameter

Type: [IFormField](#).

The new widget represents the specified form field.

### Before Parameter

Type: [sx\\_uint](#).

The new annotation will be inserted into the collection before this annotation. Can be greater than or equal to 0 and equal to annotation count.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Insert Method

The method inserts existing annotation on the page. The annotation should be from this document and shouldn't be on this page already.

### Annot Parameter

Type: [IAnnot](#).

The existing annotation.

### Before Parameter

Type: [sx\\_uint](#).

The new annotation will be inserted into the collection before this annotation. Can be greater than or equal to 0 and equal to annotation count.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Move Method

The method moves annotation in the collection order.

### Annot Parameter

Type: [IAnnot](#).

The annotation to be moved.

### Before Parameter

Type: [sx\\_uint](#).

The annotation will be moved into the collection before this annotation. Can be greater than or equal to 0 and equal to annotation count.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Remove Method

The method removes annotation from the page.

### Annot Parameter

Type: [IAnnot](#).

The annotation to be removed.



## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## ResetPopup Method

The method creates or removes popup annotation of markup annotation.

## MarkupAnnot Parameter

Type: [IAnnot](#).

The markup annotation.

## Clear Parameter

Type: [sx\\_bool](#).

If parameter is false, new popup annotation will be created, otherwise the popup will be removed.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## IAnnot Interface

The interface represents a page annotation in PDF document.

## Type Property

Type: [annot\\_type](#). Access: readonly.

The property provides the type of annotation. Using this property you can type cast the **IAnnot** interface to the derived interfaces.

## IFormFields Interface

The interface represents a form fields of PDF document.

## Count Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves number of terminal fields in the form tree.

## Root Property

Type: `IFormGroupNode`. Access: readonly.

The property retrieves root element of the form tree.

## NeedAppearances Property

Type: [sx\\_bool](#). Access: full.

The property specifying whether to construct appearances for all field widgets in the document. It should be true, if document has not provided appearances for all visible widgets.

## AutoCalculate Property

Type: [sx\\_bool](#). Access: full.

This run-time property specifying that field objects should generate [calculate event within viewer](#).

## RaiseEvents Property

Type: [sx\\_bool](#). Access: full.

This run-time property specifying that field objects should generate [field events within viewer](#).

## Item Method

The method retrieves the specified terminal field. It returns [IFormField](#) interface.

### Index Parameter

Type: [sx\\_uint](#).

The index of a field element. Can be greater than or equal to 0 and less than fields count.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Find Method

The method retrieves the terminal field by fully qualified name. It returns [IFormField](#) interface.

### sFullName Parameter

Type: [sx\\_str](#).

The fully qualified name of a field element. For a field that is the child of another field in the form tree, the fully qualified name shall be formed by appending the child field's partial name to the parent's fully qualified name, separated by a period character.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Add Method

The method adds new terminal field in the form tree. It returns [IFormField](#) interface.

### Type Parameter

Type: `field_type`.

The type of the field.

### sFullName Parameter

Type: [sx\\_str](#).

The fully qualified name of a new field element. For a field that is the child of another field in the form tree, the fully qualified name shall be formed by appending the child field's partial name to the parent's fully qualified name, separated by a period character. If form tree contains field with same fully qualified name already, this method fails.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Remove Method

The method removes the terminal field by fully qualified name. If removed field is last child of parent field, then parent field will be removed also. The method removes all widget annotations associated with this field.

### sFullName Parameter

Type: [sx\\_str](#).

The fully qualified name of a field element. For a field that is the child of another field in the form tree, the fully qualified name shall be formed by appending the child field's partial name to the parent's fully qualified name, separated by a period character.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Rename Method

The method renames the field.

### sFullName Parameter

Type: [sx\\_str](#).

The fully qualified name of a field element. For a field that is the child of another field in the form tree, the fully qualified name shall be formed by appending the child field's partial name to the parent's fully qualified name, separated by a period character.

## sNewName Parameter

Type: [sx\\_str](#).

The new name of the field. **Important!** It is not a fully qualified name.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## Copy Method

The method duplicates the field object with other name.

## sSrcFullName Parameter

Type: [sx\\_str](#).

The fully qualified name of an existing field. For a field that is the child of another field in the form tree, the fully qualified name shall be formed by appending the child field's partial name to the parent's fully qualified name, separated by a period character.

## sDestFullName Parameter

Type: [sx\\_str](#).

The fully qualified name of new field object.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## VerifySignatures Method

The method verifies all signed signatures. You must this method immediately after document opening.

---

## GetOrder Method

The method retrieves **ICalculationOrder** interface. Using this interface, you can specify order of calculation fields in the form.

---

## Calculate Method

The method throws Calculate event for the fields of form. You must handle this event to execute

recalculation of field values.

## GetDataProvider Method

The method retrieves **IFormData** interface. Using this interface, you can export or import values of the form fields in FDF, XDF or HTML formats.

## ResetValues Method

The method resets values of all fields in the form.

## ResetAppearances Method

The method regenerates or clears the appearances of all field widgets in the form.

### bClear Parameter

Type: [sx\\_bool](#).

Set this parameter to true, if you want remove existing appearances of field widgets.

## GetFonts Method

The method retrieves **IFonts** interface. Using this interface, you can explore the fonts used in the form.

## EmbedFullFonts Method

The method embeds all fonts used in the form. It returns boolean status of operation.

### bEmbPDFStd Parameter

Type: [sx\\_bool](#).

The parameter specifies to embed to standard PDF fonts, if present in form.

### bEmbWinStd Parameter

Type: [sx\\_bool](#).

The parameter specifies to embed to standard Windows fonts, if present in form.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## AddFont Method

The method inserts new font to the form.

### pFont Parameter

Type: IFont.

The font object.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## RemoveFont Method

The method removes font from the form.

### pFont Parameter

Type: IFont.

The font object.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## IFormField Interface

The interface represents a form field element of PDF document.

## IStatusPDFA Interface

The callback interface, you need to implement it for using in ConformToPDFA method for monitoring of converting process.

### OnMessage

The method is called everytime when [ConformToPDFA](#) procedure needs to inform your application about error, warning or other event in process of converting. After first error message, ConformToPDFA procedure throws an exception and breaks converting. The method must return **true** to continue

converting or **false** to break ConformToPDFA procedure.

## Message Parameter

Type: [IMessagePDFA](#).

The converting message.

## IMessagePDFA Interface

The interface represents a converting message for [procedure of PDF/A converting](#). The message provides category of message (error, warning, information) and coded description of a message.

## Type Property

Type: [sx\\_flags](#). Access: readonly.

The property provides the category and description of message. Both values are “packed” in this property, as set of bit flags.

## Message Category

Bits 24 through 31 of Type property contain the message category. Use bitwise AND operation between Type property and **PDF::pdfa\_mess\_cat** mask to get category of message, it could be:

- pdfa\_mess\_info - this message informs your application about performing of converting
- pdfa\_mess\_warn - this message informs your application about modifications in the document to make it PDF/A conform
- pdfa\_mess\_error - this message informs your application about finding non-conform and non-convertable element or missing mandatory element in the processed document, after this message the converting procedure will be broken with exception.

## Message Description

Bits 0 through 23 of Type property contain the message description. Use bitwise AND operation between Type property and **PDF::pdfa\_mess\_type** mask to get coded description of message, it could be:

- pdfa\_mess\_begin - information, the procedure started
- pdfa\_mess\_end - information, the procedure successfully performed
- pdfa\_mess\_security - error, the document must be non-encrypted
- pdfa\_mess\_version - warning, format version was adjusted

- pdfa\_mess\_doc\_id - warning, document ID was generated
- pdfa\_mess\_doc\_actions - warning, document actions was removed
- pdfa\_mess\_optional\_content - warning, optional content groups was removed, the optional content transformed to normal content
- pdfa\_mess\_need\_appearances - warning, the form property "NeedAppearances" was cleared
- pdfa\_mess\_field\_actions - warning, form field actions was removed
- pdfa\_mess\_annot\_appearance - warning, the annotation appearance was adjusted
- pdfa\_mess\_portfolio - error, can't convert the portfolio to PDF/A
- pdfa\_mess\_emb\_files - warning, the embedded file(s) was removed
- pdfa\_mess\_xfa - error (dynamic XFA isn't allowed) or warning ("static" XFA removed)
- pdfa\_mess\_mark\_info - warning, the document mark info was adjusted
- pdfa\_mess\_java\_script - warning, the document JavaScript was removed
- pdfa\_mess\_lang - warning, the document language identifier was adjusted
- pdfa\_mess\_struct\_tree - warning, the logical structure tree of a document was adjusted
- pdfa\_mess\_output\_intents - error (can't find standard color profile) or warning (standard color profile was embedded)
- pdfa\_mess\_annot\_type - warning, annotation of non-allowed type was removed
- pdfa\_mess\_action\_type - warning, event action of non-allowed type was removed
- pdfa\_mess\_widget\_actions - warning, widget action was removed
- pdfa\_mess\_annot\_opacity - warning, annotation opacity was cleared
- pdfa\_mess\_annot\_flags - warning, annotation flags was adjusted
- pdfa\_mess\_page\_actions - warning, page action was removed
- pdfa\_mess\_image\_alternates - warning, alternate images was removed
- pdfa\_mess\_image\_opi - warning, image OPI entry was removed
- pdfa\_mess\_image\_interpolate - warning, image interpolation property was cleared
- pdfa\_mess\_rendering\_intent - warning, image rendering intent was cleared
- pdfa\_mess\_font\_embedding - error (non-embedded font can't be embedded) or warning (nonembedded font was embedded)



- pdfa\_mess\_xobject\_opi - warning, composite image OPI entry was removed
- pdfa\_mess\_xobject\_ps - warning, PostScript image was removed
- pdfa\_mess\_xobject\_ref - warning, reference image was removed
- pdfa\_mess\_smask - warning, soft mask (mask image) was removed
- pdfa\_mess\_transparency - warning, transparency property was cleared
- pdfa\_mess\_jpx\_compression - warning, JPEG2000 compression was replaced with JPEG
- pdfa\_mess\_lzw\_compression - warning, LZW compression was replaced with Flate
- pdfa\_mess\_crypt\_filter - error, crypto filter isn't allowed
- pdfa\_mess\_external\_stream - warning, reference to an external stream was removed
- pdfa\_mess\_perms - warning, document permissions was adjusted
- pdfa\_mess\_blend\_mode - warning, blend mode was removed
- pdfa\_mess\_glyph\_widths - warning, glyph width was adjusted
- pdfa\_mess\_font\_cidset - warning, CID set was adjusted or removed
- pdfa\_mess\_font\_tounicode - error, missed a mandatory unicode map of a font
- pdfa\_mess\_page\_userunit - error or warning, page user unit was cleared
- pdfa\_mess\_missing\_glyph - error, missed a mandatory glyph of a character
- pdfa\_mess\_struct\_type - warning, type of the document logical structure was adjusted
- pdfa\_mess\_struct\_lang - warning, language ID of the document logical structure was adjusted
- pdfa\_mess\_embedded\_files - warning, embedded files was removed
- pdfa\_mess\_renditions - warning, renditions was removed
- pdfa\_mess\_alternate\_presentations - warning, alternate presentations was removed
- pdfa\_mess\_page\_pressteps - warning, sub-page navigation was removed
- pdfa\_mess\_file\_association - warning, embedded files was adjusted

## Enumerations

### Compact\_objects Enumeration

The members of this enumeration are the options for compact operation of the page. You can combine

the enumeration values by using the bitwise OR operator.

### **compact\_objects\_content**

The page content must be compacted.

### **compact\_objects\_composite\_images**

The composite images of the page must be compacted.

### **compact\_objects\_raster\_images**

The raster images of the page must be compacted.

### **compact\_objects\_fonts**

The fonts used by the page must be compacted.

---

## **Annot\_type Enumeration**

The members of the enumeration are the possible types of PDF annotations.

**annot\_type\_undefined**

**annot\_type\_text**

**annot\_type\_link**

**annot\_type\_freertext**

**annot\_type\_line**

**annot\_type\_square**

**annot\_type\_circle**

**annot\_type\_polygon**

**annot\_type\_polyline**

**annot\_type\_highlight**

**annot\_type\_underline**

**annot\_type\_squiggly**

**annot\_type\_strikeout**

**annot\_type\_caret**

**annot\_type\_stamp**

**annot\_type\_ink**

**annot\_type\_popup**

annot\_type\_attachment  
annot\_type\_sound  
annot\_type\_movie  
annot\_type\_screen  
annot\_type\_widget  
annot\_type\_printermark  
annot\_type\_trapnet  
annot\_type\_watermark  
annot\_type\_3D  
annot\_type\_redact  
annot\_type\_projectio  
annot\_type\_richmedia  
annot\_type\_reply\_to

# SX::XPS Namespace

## IPackage Interface

The interfaces provides interface of XPS package object implemented by SDK. This interface is derived from [IBaseDocument](#). You can create new instance of XPS package object using [IAppFactory::CreateXPSPackage](#) method.

## SX::Content Namespace

This namespace covers the Rich Content API (RC) that provides a close access to page content of PDF or XPS documents. Using RC interfaces and methods you can create content of new pages (fill the blank pages), edit content of pages created by any other software. Also you can explore content of pages, for such tasks as: recognize placement content objects (text, images, vector elements) on the page, format of text, etc.

The RC API represents page content as a hierarchical structure of content objects. The PDF and XPS documents store the page content in different own formats and both formats are not suitable for direct using (creating / editing / exploring) of content object. The library translates the internal structures of page content in the hierarchical structure of content objects independently from native format of content. If the RC structure (or objects) was modified, the library makes backward translation (from RC to native format) automatically.

The key interface of RC API is the [IRichContent interface, see more details below](#).

## Coordinate System and Unit of Measure

The unified coordinate system of RC is traditional for the Windows. The unit of measure is the point (1/72 of an inch). The x-coordinates increase to the right; y-coordinates increase from top to bottom.

The content owners (page or composite image) in the PDF or XPS documents have own native coordinate systems, these systems are different and don't coincide with this unified coordinate system. You can transform the coordinates between RC and native systems using [GetMatrix method](#) of [IRichContent](#) interface.

## IRichContent Interface

The interface provides direct access to the page content (or content of composite image) as a hierarchical structure of content objects. The structure of content and content objects are imported from native content of PDF or XPS documents because native format of content in these standards is different and optimized for displaying or printing of content. PDF Xpansion SDK imports native content and builds Rich Content structure for every requested page or composite image automatically, if it is not builded yet. In the case of empty page or composite image this interface is fully functional but the collection of content objects is initially empty.

If you want to modify or create new content of page or composite image, you need request Rich Content (call GetRichContent) than change structure of Rich Content or properties of Rich Content objects and **release all references to IRichContent interface** – exactly at this timepoint the PDF Xpansion SDK exports Rich Content structure to native content and replaces old content of appropriate page or composite image.

**Important!** Before export of Rich Content to native content, any changes of Rich Content are not “visible” in PDF or XPS document.

**Windows:** We highly recommend you use an instance of an [ObjPtr template class](#) where you can use a **Content::IRichContent** pointer in a safe manner.

### DotNET:

At this platform IRichContent is inherited from [System.IDisposable interface](#). If you make any changes in Rich Content and you want apply these changes (export to native content) immediately, you need to use **Dispose** method, because even you have not any references at this Rich Content, the export will be executed with delay.

If you want process multiple pages using IRichContent, we recommend you release IRichContent of processed page before you request IRichContent of other page.

---

## Objects Property

Type: [IRichCollection](#). Access: readonly.

The property provides first level collection of the content objects.

---

## Modified Property

Type: [sx\\_bool](#).

The property indicates that the content has been modified.

---

## GetUPI Method

The method retrieves “units per inch” property. Is a measure of units in the native coordinate system of content owner (page or self-contained composite image), in particular the number of units that can be placed within the span of 1 inch.

---

## GetMatrix Method

The method retrieves matrix of difference between coordinate system of content owner (page or self-contained composite image) and [unified coordinate system of RC](#). Using this matrix you can transform any coordinates from the page space to the RC space and vice versa (using inverted matrix). It works as the [PDF::IPage::CalcMatrix](#) method.

In other words inversion of this matrix gives you transformation to the “coordinate system” of first level collection (see Objects property above) which is exactly native page space.

## M Parameter

Type: [sx\\_matrix](#).

The transformation matrix, it is output parameter.

## Clear Method

The method removes all content objects in first level collection.

## CreateBrush Method

The method creates a new brush object. It returns IBrush interface.

**Important:** you may not use this object with the content objects which are not childs of this IRichContent.

### Type Parameter

Type: brush\_type.

The type of the brush.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## CreateSolidBrush Method

The method creates a new solid brush object. It returns ISolidBrush interface.

**Important:** you may not use this object with the content objects which are not childs of this IRichContent.

### Color Parameter

Type: [sx\\_color](#).

The color of the brush.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## CreateFont Method

The method creates a new font object based at the system font. It returns IFont interface.

**Important:** you may not use this object with the content objects which are not childs of this IRichContent.

### Family Parameter

Type: [sx\\_str](#).

The font family name.

## Weight Parameter

Type: [sx\\_uint](#).

The density of a typeface, use 400 for regular density and 700 for bold density.

## Italic Parameter

Type: [sx\\_bool](#).

The style of a font, italic or normal.

## Stretch Parameter

Type: [sx\\_uint](#).

The degree to which a font has been stretched compared to a font's normal aspect ratio, use 5 for normal, 3 for condensed and 7 for expanded styles of a font.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## CreatePath Method

The method creates a new geometry path object. It returns IPath interface.

**Important:** you may not use this object with the content objects which are not childs of this IRichContent.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## CreatePen Method

The method creates a new pen object. It returns IPen interface.

**Important:** you may not use this object with the content objects which are not childs of this IRichContent.

## Width Parameter

Type: [sx\\_double](#).

The width of pen, in points.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.



## CreateTextStyle Method

The method creates a new text style object. It returns ITextStyle interface.

**Important:** you may not use this object with the content objects which are not childs of this IRichContent.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## IRichCollection Interface

This interface represents collection of content objects in a hierarchical structure on a “horizontal” level. The order of objects in a collection explicitly defines the z-order of the order. When objects overlap, z-order determines which one covers the other. An object with a larger index within a collection generally covers an object with a lower one.

The coordinate system for the first level collection is native page space (see [Content::IRichContent::GetMatrix method](#)), for the second and higher levels it is “coordinate system” of group content object (see [Content::IRichObject::GetTransformation method](#)).

## Root Property

Type: [IRichContent](#). Access: readonly.

The property retrieves interface which represents current hierarchical structure of content objects.

## Group Property

Type: [IRichGroup](#). Access: readonly.

The property retrieves an immediate parent group object for the collections of second or higher levels. For the first level collection this property is null.

## Count Property

Type: [sx\\_uint](#). Access: readonly.

The property retrieves number of objects in the collection.

## Item Method

The method retrieves the specified object. It returns [IRichObject](#) interface.

### Index Parameter

Type: [sx\\_uint](#).

The index of an object. Can be greater than or equal to 0 and less than number of objects.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## IndexOf Method

The method searches for the specified object and returns the zero-based index of an object within the collection.

## Page Parameter

Type: [IRichObject](#).

The content object.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## Insert Method

The method creates new content object of specified type and inserts it to the specified position. It returns [IRichObject](#) interface.

## To Parameter

Type: [sx\\_uint](#).

The position to insert an object. Can be greater than or equal to 0 and less than or equal to number of objects.

## Type Parameter

Type: rich\_type.

The type of content object.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## InsertCopy Method

The method clones a specified object and inserts new copy to the specified position. It returns [IRichObject](#) interface. If a copied object uses shareable resource objects, they will be used by reference. For example if you copy one text object to another and then you change text style object of new object, the original text object will use a changed style also. The child content objects of group object are copied with parent object.

**Important.** The object to be copied must be located in the any collection within a hierarchical structure where current collection is located, in other words both collections must reference to one [IRichContent](#) object (see Root property).

## To Parameter

Type: [sx\\_uint](#).

The position to insert an object. Can be greater than or equal to 0 and less than or equal to number of objects.

## Object Parameter

Type: [IRichObject](#).

The content object to be copied.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Move Method

The method moves an content object to the specified position within collection. In other words this method changes z-order of objects.

## From Parameter

Type: [sx\\_uint](#).

The index of an object to move. Can be greater than or equal to 0 and less than number of objects.

## To Parameter

Type: [sx\\_uint](#).

The position to insert an object. Can be greater than or equal to 0 and less than or equal to number of objects.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Remove Method

The method removes a specified content object from the collection and a hierarchical structure.

## Index Parameter

Type: [sx\\_uint](#).

The index of an object. Can be greater than or equal to 0 and less than number of objects.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

# IRichObject Interface

The interfaces declares common properties (and functionality) for all types of content objects supported by Rich Content API.

## Type Property

Type: `rich_type`. Access: readonly.

The property retrieves type of content object. Using this property you can type cast the **IRichObject** interface to the derived interfaces.

## Collection Property

Type: [IRichCollection](#). Access: readonly.

The property retrieves an owner collection of this object. Use this collection to get `IRichContent` or `IRichGroup` of for this object (see `IRichCollection::Root` and `IRichCollection::Group` properties).

## GetMatrix Method

The method retrieves the matrix which defines transformation of object within [the coordinate system of owner collection](#). If you need the matrix of transformation within [unified coordinate system of RC](#) you must use **GetTransformation** method.

## M Parameter

Type: [sx\\_matrix](#).

The transformation matrix, it is output parameter.

## SetMatrix Method

The method defines transformation of object within [the coordinate system of owner collection](#). If you want define the transformation within [unified coordinate system of RC](#) you must use **ApplyTransformation** method.

## M Parameter

Type: [sx\\_matrix](#).

The transformation matrix.

---

## GetTransformation Method

The method retrieves the matrix which defines transformation of object within [unified coordinate system of RC](#). If you need the individual matrix of object (transformation within [the coordinate system of owner collection](#)) you must use **GetMatrix** method.

### M Parameter

Type: [sx\\_matrix](#).

The transformation matrix, it is output parameter.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## ApplyTransformation Method

The method defines transformation of object within [unified coordinate system of RC](#). If you want define the individual matrix of object (transformation within [the coordinate system of owner collection](#)) you must use **SetMatrix** method.

### M Parameter

Type: [sx\\_matrix](#).

The transformation matrix.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## GetBBox Method

The method retrieves the bounding box of the content object within the coordinate system specified by transformation matrix (see first parameter).

### M Parameter

Type: [sx\\_matrix](#).

The transformation matrix specifies the coordinate system where you want to get bounding box of object. Use identity matrix to get original dimensions of object (position is irrelevant in this case) or use matrix returned by **GetTransformation** method to get bounding box within [unified coordinate system of RC](#).

### Rc Parameter

Type: [sx\\_rect](#).

The bounding box of content object, it is output parameter.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

# SX::XMP Namespace

Metadata is data that describes the characteristics or properties of a document. It can be distinguished from the main contents of a document. For example, for a word processing document, the contents include the actual text data and formatting information, while the metadata might include such properties as author, modification date, or copyright status. Metadata in PDF documents may be specified for the document itself or for individual components of a PDF document (pages, fonts, images).

Metadata of PDF document is in XML based XMP format (Extensible Metadata Platform). This format standardizes the definition, creation, and processing of metadata. XMP metadata may include properties from one or more of the schemas. An XMP schema is a set of top level property names in a common XML namespace, along with data type and descriptive information. Typically, an XMP schema contains properties that are relevant for particular types of documents or for certain stages of a workflow. PDF and PDF/A formats define some schemas for use in a PDF document.

PDF Xpansion SDK provides XMP API for processing metadata within predefined schemas used in PDF documents. The starting point of XMP API is XMP::IDocument interface, see it below.

## IDocument Interface

The interface represents a metadata object implemented by SDK. You can create new instance of document object using [IAppFactory](#) interface.

**Windows:** This interface is derived from [IRefObject](#) at this platform.

## OpenStream Method

The method loads the metadata from XML stream. It returns [validation status](#) of metadata. The metadata can be used if status of loading is **xmp\_status\_xmp**.

### Stream Parameter

Type: [ISequentialStream](#).

The XML formatted metadata stream. Please reset stream before load.

### Profile Parameter

Type: [xmp\\_profile](#).

The type of validation profile.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## OpenFromString Method

The method loads the metadata from XML string.. It returns [validation status](#) of metadata. The metadata can be used if status of loading is **xmp\_status\_xmp**.

## XMP Parameter

Type: [sx\\_str](#).

The string which contains the XML formatted metadata. You can get this string from PDF document using PDF::IDocProperties::GetMetadata method.

## Profile Parameter

Type: [xmp\\_profile](#).

The type of validation profile.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## SaveAsStream Method

The method saves the XML formatted metadata to the stream.

## Stream Parameter

Type: [ISequentialStream](#).

The stream for save. Please reset stream before save.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## SaveToString Method

The method saves the XML formatted metadata to the string. It returns [IStr](#) object. You can use this string for changing of metadata in PDF document, see PDF::IDocProperties::SetMetadata method.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.



---

## GetScheme Method

The method retrieves interface of specified scheme. It returns base interface IScheme, you can type cast this interface to the derived interface according to requested scheme.

### Scheme Parameter

Type: xmp\_scheme.

The type of requested scheme.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## Reset Method

The method resets the metadata document - remove all schemas (if present) and creates root structure of metadata.

---

## Enumerations

---

### xmp\_profile enumeration

The enumeration declares some validation profiles. The profile defines set of schemas which can be used in the metadata.

---

### xmp\_status enumeration

The enumeration declares possible results of metadata validation.

# SX::EInvoice Namespace

## PDF/A-3 based invoice

### Create new invoice

For the creation of electronic invoices, you need a PDF file that represents your invoice in the traditional (human readable) form and XML file that represents structured invoice data and should comply with the desired standard: XRechnung, ZUGFeRD or FACTUR-X. If you do not have a ready PDF file as invoice, you can [create it "on the fly" from text and/or images](#). At first, you need to call **CreateFromPDF** or **Create** methods for loading and validation of the XML file. Please note that validation of XML at this step is very simple, it checks:

- all required namespaces of XML invoice
- conformity of root node
- conformity of invoice profile definition

The **CreateFromPDF** method performs the conversion of the source PDF file to PDF/A-3 as well. Please note that not all PDF files can be converted to PDF/A-3, it is unlikely, but possible – in this case the method throws an error. If you used the **Create** method, you must add one or more pages using the **CreatePage** method and [build page content](#).

After the successful creation of the invoice document with embedded XML, you can use the **Save** method to save it to the file.

If you do not have a ready XML-formatted invoice, you can use our [IInvoiceData](#) interface to produce such invoice "on the fly".

### Validate and read data from existing invoice

If you get a PDF file, supposedly with an electronic invoice, you need to detect and validate the conformity of this file to one of the invoice standards: XRechnung, ZUGFeRD or FACTUR-X. Call the **Open** method for loading and validation of the PDF invoice. Validation at this step includes:

- conformity of PDF metadata to PDF/A-3 and invoice standards
- embedding conformity of XML invoice
- optional and simple check of XML invoice (required namespaces, root node, profile definition)
- optional validation of complete PDF/A-3 conformity

If validation was successfully done, you can get standard and profile of the invoice using **Standard** and **Profile** properties. If you want to process the embedded XML data, call the **GetInvoice** method to get

the XML invoice file. Use our [IInvoiceData](#) interface for extracting invoice data from XML file. Optionally you can load and display PDF representation of the invoice using our [PDF viewer](#).

## Build PDF representation of invoice

If you do not have a ready PDF file with an invoice, you need to use the **Create** method to create an empty PDF document and call the **CreatePage** method at least once for adding an empty page to this PDF document. Using **IPage** interface returned by this method, you can place the content on the added page. There are two possible scenarios:

1. You have (a) scanned page(s) of the paper invoice (or raster image(s) with content of invoice pages) – in this case you place one image on each added page of the PDF document
2. You have the invoice data only – in this case you need to place many text elements on the page to layout the visual representation of the invoice data. You can also place raster images (company logo, etc.) and vector lines at the page

## InvoiceDocument Interface

The interface provides functionality for processing PDF/A-3 based invoices: XRechnung, ZUGFeRD or FACTUR-X. A PDF/A-3 based invoice combines the visual representation of an invoice (one or more pages of the PDF file), plus structured invoice data for further automated processing (XML formatted data must be embedded in the PDF file). This interface is derived from [IBaseDocument](#), so you can load it to our [PDF viewer](#). You can create a new instance of an object using the [IAppFactory](#) interface.

**Windows:** We highly recommend you use an instance of an [ObjPtr template class](#) where you can use an `Invoice::IInvoiceDocument` pointer in a safe manner.

### Title Property

Type: [IStr](#). Access: full.

The property specifies the title of document (within PDF part and metadata).

### Subject Property

Type: [IStr](#). Access: full.

The property specifies the subject of document (within PDF part and metadata).

### Author Property

Type: [IStr](#). Access: full.

The property specifies the author (person or company) of document (within PDF part and metadata).

## Creator Property

Type: [IStr](#). Access: full.

The property specifies the creator (software) of document (within PDF part and metadata).

## Standard Property

Type: [einv\\_standard](#). Access: readonly.

The property retrieves standard of invoice - XRechnung, ZUGFeRD or FACTUR-X.

## Profile Property

Type: [einv\\_profile](#). Access: readonly.

The property retrieves profile of invoice - MINIMUM, BASIC WL, BASIC, EN16931 (COMFORT for ZUGFeRD 1.0) or EXTENDED.

## Open Method

The method loads the PDF/A-3 document from stream or file. It returns [validation status](#) of loaded document, [see more information loading validation here](#).

## Stream Parameter

Type: [ISequentialStream](#).

The document data stream. Please reset stream before load.

## Opt Parameter

Type: `sx_flags`, see also [einv\\_options](#) enumeration.

The parameter specifies the options of opening. You can combine the enumeration values by using the bitwise OR operator.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Create Method

The method [creates empty PDF/A-3 based invoice](#) and embeds an XML invoice. The standard of invoice must be correctly declared using **opt** parameter. At least [one PDF page must be added](#) before saving a document.

## pInvoice Parameter

Type: [ISequentialStream](#).

The XML invoice stream or file. Please reset stream before load.

## Opt Parameter

Type: `sx_flags`, see also [einv\\_options](#) enumeration.

The parameter declares standard of XML invoice and other options of creating.

If the declared standard does not correspond to the standard XML invoice, then method throws an error.

The `einv_options_def` value specifies ZUGFeRD 1.0 invoice (it is outdated standard).

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## CreateFromPDF Method

The method [creates PDF/A-3 based invoice from PDF stream or file](#) and embeds an XML invoice. The standard of invoice must be correctly declared using **opt** parameter. Please note that not all PDF files can be converted to PDF/A-3, it is unlikely, but possible – in this case the method throws an error.

## pInvoice Parameter

Type: [ISequentialStream](#).

The XML invoice stream or file. Please reset stream before load.

## pPDF Parameter

Type: [ISequentialStream](#).

The PDF document data stream. Please reset stream before load.

## Opt Parameter

Type: `sx_flags`, see also [einv\\_options](#) enumeration.

The parameter declares standard of XML invoice and other options of creating.

If the declared standard does not correspond to the standard XML invoice, then method throws an error.

The `einv_options_def` value specifies ZUGFeRD 1.0 invoice (it is outdated standard).

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## CreatePage Method

The method inserts new [empty page](#) to the PDF/A-3 based invoice. It returns **IPage** interface for building content of page.

### W Parameter

Type: [sx\\_double](#).

The page width in points.

### H Parameter

Type: [sx\\_double](#).

The page height in points.

## Save Method

The method saves the invoice document to the stream or file.

### Stream Parameter

Type: [ISequentialStream](#).

The stream for save.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## GetInvoice

The method extracts stream with XML invoice from opened PDF/A invoice.

### Stream Parameter

Type: [ISequentialStream](#).

The stream for XML invoice.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## GetAttachment

The method extracts stream with specified embedded file from opened PDF/A invoice.

### Stream Parameter

Type: [ISequentialStream](#).

The stream for attachment.

## pFilename Parameter

Type: [sx\\_str](#).

The filename of attachment.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## SetAttachment

The method embeds specified stream with additional document or image to created PDF/A invoice.

## Stream Parameter

Type: [ISequentialStream](#).

The stream with attachment.

## pFilename Parameter

Type: [sx\\_str](#).

The filename of attachment.

## pMime Parameter

Type: [sx\\_str](#).

The MIME type of attachment.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## CreateInvoice

The method creates new object for processing XML based invoices, it returns [IInvoiceData](#) interface.

**Windows:** We highly recommend you use an instance of an [ObjPtr template class](#) where you can use an `ElInvoice::IInvoiceData` pointer in a safe manner.

## InvoiceData Interface

The interface provides functionality for processing XML based invoices: XRechnung, ZUGFeRD or FACTUR-X. You can create a new instance of this object using the [InvoiceDocument::CreateInvoice](#) method.

**Windows:** We highly recommend you use an instance of an [ObjPtr template class](#) where you can use an `Invoice::InvoiceData` pointer in a safe manner.

---

## Standard Property

Type: [inv\\_standard](#). Access: readonly.

The property retrieves standard of invoice - XRechnung, ZUGFeRD or FACTUR-X.

---

## Profile Property

Type: [inv\\_profile](#). Access: readonly.

The property retrieves profile of invoice - MINIMUM, BASIC WL, BASIC, EN16931 (COMFORT for ZUGFeRD 1.0) or EXTENDED.

---

## Test Property

Type: [sx\\_bool](#). Access: full, for opened invoices access is readonly.

The property defines an invoice as test invoice. It can be set within extended profile only (ZUGFeRD 1.0 allows it for all profiles).

---

## BusinessProcess Property

Type: [sx\\_str](#). Access: full, for opened invoices access is readonly.

The property defines the business process identifier of invoice. It can be set within all profiles (ZUGFeRD 1.0 allows it for extended profile only).

---

## Description Property

Type: [InvoiceDescription](#). Access: readonly.

The property retrieves interface which provides description properties of invoice – general properties for the whole document such as document number, document title, type code, issue date, notes, etc.

---

## Transaction Property

Type: [InvoiceTransaction](#). Access: readonly.

The property retrieves interface which represents main parts of invoice – structured information about all transactions in a document: individual invoice positions, information about goods and services, prices, reference products, etc.



## Open Method

The method loads the invoice data from XML stream or file. It returns [validation status](#) of loaded invoice. It checks complete XML structure - required namespaces and elements. More detailed information about validation of XML structure you can get using callback interface **IIInvoiceValidation**. Set your handler using **SetValidationHandler** method before loading of invoice data.

You can not use properties and methods of **IIInvoiceData** interface, if this method returns status other than "einv\_status\_valid". If method returns "einv\_status\_err\_xml\_inv" status, the XML contains invoice, but this invoice is incomplete, some elements missed. You can load available data of such invoice using **ImportData** method. If method returns "einv\_status\_ubl\_syntax" status, the XML contains invoice in UBL syntax, you can import invoice data from this XML using **ImportUBL** method.

### Stream Parameter

Type: [ISequentialStream](#).

The invoice data stream. Please reset stream before load.

### St Parameter

Type: [einv\\_standard](#).

The parameter specifies expected standard of invoice. If invoice corresponds to other standard, then method throws an error. You can specify "einv\_standard\_unknown" value for opening any supported standards, then actual standard of loaded invoice you can get using **Standard** property.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## ImportData Method

The method loads the invoice data from XML stream or file. It is similar to the **Open** method, but with one difference - you can use imported data even if invoice is incomplete and method returns "einv\_status\_err\_xml\_inv" status. If method returns "einv\_status\_ubl\_syntax" status, the XML contains invoice in UBL syntax and you can import invoice data using **ImportUBL** method or you can specify "einv\_options\_allow\_ubl" and **ImportData** will import it automatically.

### Stream Parameter

Type: [ISequentialStream](#).

The invoice data stream. Please reset stream before load.

### Opt Parameter

Type: `sx_flags`.

The parameter is reserved.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## ImportUBL Method

The method imports the invoice data from XML stream or file that uses UBL syntax according to XRechnung standard. It returns [validation status](#) of imported invoice. If import status is `einv_status_valid` you can get invoice data as for any opened XML invoice. Also, you can save imported invoice in UN/CEFACT syntax using **Save** method. Please note, you cannot save a noncomplete invoice (if one or more required elements are missing). You can check completeness of invoice by calling **CheckRequirements** method before saving.

## Stream Parameter

Type: [ISequentialStream](#).

The invoice data stream. Please reset stream before load.

## Opt Parameter

Type: `sx_flags`.

The parameter is reserved.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## Create Method

The method creates new empty invoice. Fill properties of new invoice, then call **Save** method. Please note, you cannot save a noncomplete invoice (if one or more required elements are missing). You can check completeness of invoice by calling **CheckRequirements** method before saving.

## St Parameter

Type: [einv\\_standard](#).

The parameter specifies the necessary standard of invoice.

## Pr Parameter

Type: [einv\\_profile](#).

The parameter specifies the necessary profile of invoice. Please note, some standards allow not all profiles.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Save Method

The method saves the invoice data to the XML stream or file. Please note, you cannot save a noncomplete invoice (if one or more required elements are missing). You can check completeness of invoice by calling **CheckRequirements** method before saving.

### Stream Parameter

Type: [ISequentialStream](#).

The stream for saving of XML invoice.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## CheckRequirements Method

The method checks completeness of invoice before saving. It returns false, if invoice is noncomplete and you cannot call **Save** method. More detailed information about missed elements you can get using callback interface **InvoiceValidation**. Set your handler using **SetValidationHandler** method before calling this method.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## Adjust Method

The method transforms XML from one standard and profile to other standard and/or profile. Before calling this method, an existing XML invoice must be successfully opened using **Open** method. The standard can be changed without restrictions, the new profile can be same or upgraded, profile degradation is restricted.

**Important!** The ZUGFeRD 1.0 standard in comparison with other standards has much less required elements. Therefore, after transforming ZUGFeRD 1.0 invoice to any other standard, it is highly possible that transformed invoice is noncomplete according to new standard and missing elements must be added before saving. You can check completeness of invoice by calling **CheckRequirements** method in this case.

After transforming to ZUGFeRD 1.0 from any other standard, an incomplete state of invoice is possible also, but in extremely rare cases.

### St Parameter

Type: [einv\\_standard](#).

The parameter specifies the new standard of invoice.

## Pr Parameter

Type: [einv\\_profile](#).

The parameter specifies the new profile of invoice. Please note, some standards allow not all profiles.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

---

## ExportPDF Method

The method allows to visualize the invoice data as PDF file. Invoice must be opened, imported or created before calling this method.

## pTemplate Parameter

Type: [ISequentialStream](#).

The stream with design template for invoice formatting. Please ask us for test templates or creation of customized templates.

## pPDF Parameter

Type: [ISequentialStream](#).

The stream for saving of builded PDF with invoice.

## Opt Parameter

Type: `sx_flags`.

The parameter is reserved.

---

## SetValidationHandler Method

The method sets callback handler for processing detailed information about missed elements of invoices or other problems detected by checking of standard requirements.

## pHandler Parameter

Type: `IIInvoiceValidation`.

The callback handler. The `IIInvoiceValidation` interface must implemented by client application.

---

## IIInvoiceDescription Interface

This interface provides a description of invoice properties – general properties for the whole document such as document number, document title, type code, issue date, notes, etc.

**Important!** This interface and all subinterfaces allow to change the created (or imported) invoices only! For opened invoices all properties are readonly and "c" parameter (create new object) of all methods must be false.

## ID Property

Type: [sx\\_str](#). Access: full.

A law specified sequential number to uniquely identify the invoice within the seller's records, based on one or more series. It can be based on one or more series of numbers, and may contain alphanumeric characters. No identification scheme is to be used.

## Name Property

Type: [sx\\_str](#). Access: full.

Free text to identify the document type, e.g. "Invoice". It can be set within extended profile only (ZUGFeRD 1.0 allows it for all profiles).

## TypeCode Property

Type: [sx\\_uint](#). Access: full.

Document code type according to the UNTDID (United Trade Data Interchange Directory) 1001. Most typical code examples: 380 (commercial invoice), 381 (credit note), 384 (corrected invoice), 386 (advance payment invoice).

## IssueTime Property

Type: [IDateTime](#). Access: full.

Invoicing date. It contains of the date and time value and format template string according to the regional used formats.

## Copy Property

Type: [sx\\_bool](#). Access: full.

Indicator Original / Copy. Should be set to **true** if the recipient explicitly requests to be resent. The property is only required if it is not a multi-piece delivery with the same content. It can be set within extended profile only (ZUGFeRD 1.0 allows it for all profiles).

## Language Property

Type: **LangCode2**. Access: full.

Language designation within Alpha-2 code space (ISO 639-1). For example, LangCode2\_de. It can be set within extended profile only (ZUGFeRD 1.0 allows it for all profiles).

## GetNote Method

Detailed information as a code and a free text to a whole document. Bilateral agreed text items in form of a code or a free text. The method returns [INote](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

### i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be greater than or equal to 0 and less than number of objects. If c parameter is true, then index parameter can be equal to number of objects for creating of new element.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## GetEffectivePeriod Method

Contractual due date of the invoice. Information only required if the contractual due date differs from the due date of the payment (e.g. for SEPA direct debits). The method returns [ITimePeriod](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

### Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

## InvoiceTransaction Interface

The interface represents main parts of invoice. It contains the structured information about all transactions in a document: individual invoice positions, information about goods and services, prices, reference products, etc.

**Important!** This interface and all subinterfaces allow to change the created (or imported) invoices only! For opened invoices all properties are readonly and "c" parameter (create new object) of all methods must be false.

---

## Agreement Property

Type: [IAgreement](#). Access: readonly.

Detailed information on the prices of goods and services invoiced. Information about the order, the corresponding agreement and additional documents.

---

## Delivery Property

Type: [IDelivery](#). Access: readonly.

Delivery properties for a position. Billed quantity and free quantity of goods and/or services. Detailed information on the different goods recipient and final goods recipient. Detailed information on the delivery event.

---

## Settlement Property

Type: [ISettlement](#). Access: readonly.

Detailed information on taxation, extra charges and discounts, billing period, for a position. Accounting references of the buyer.

---

## GetLineItem Method

Detailed information on an invoice item: item number and codes, free text to the item, product information and classification, country of origin, information about the included products. The method returns [ILineItem](#) interface. At least one line item must be specified for any valid invoice (excluding MINIMUM and BASIC\_WL profiles).

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

### i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be greater than or equal to 0 and less than number of objects. If `c` parameter is true, then index parameter can be equal to number of objects for creating of new element.

## Errors

The method produces the [exceptions](#) if failed, see [ErrorCode](#) property of exception for details.

# IAgreement Interface

This interface represents detailed information about:

- The seller and the buyer: identification, addresse, contacts for communication, tax identification
- Seller tax representative if available: trade contact (person, department and contact data)
- Product end user
- Information about delivery conditions
- Detailed information on the order and order confirmation
- Detailed information on the appropriate agreement

## BuyerReference Property

Type: [sx\\_str](#). Access: full.

An identification determined and assigned by the buyer and used for internal routing, e.g. contact details, department, office ID, project code. This identification should be specified by the seler in the invoice.

## GetSeller Method

Detailed information on the seller: seller ID and global ID, name, description as a legal information, details to the organization, trading business name, trade contact, tax registration. This part must be specified for any valid invoice (excluding MINIMUM and BASIC\_WL profiles). The method returns [IParty](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetBuyer Method

Detailed information on the buyer: buyer ID and global ID, name, description as a legal information, details to the organization, trading business name, trade contact, tax registration. This part must be specified for any valid invoice (excluding MINIMUM and BASIC\_WL profiles). The method returns [IParty](#) interface.



## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetSellerTaxRepresentative Method

Detailed information on the tax representative of the seller: ID and global ID, name, description as a legal information, details to the organization, trading business name, trade contact, tax registration. This part is restricted for ZUGFeRD 1.0. The method returns [IParty](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetEndUser Method

Detailed information on the product end user: end user ID and global ID, name, description as a legal information, details to the organization, trading business name, trade contact, tax registration. This part can be set within extended profile only. The method returns [IParty](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetDeliveryTerms Method

Delivery terms type code. This part can be set within extended profile only. This part can be set within extended profile only. The method returns [IDeliveryTerms](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetSellerOrder Method

Detailed information on the order referenced document. This part is restricted for ZUGFeRD 1.0. That is the order number and the order date with a format template string.

The method returns [IRefDoc](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetBuyerOrder Method

Detailed information on the buyer order referenced document. That is the order number and the order date with a format template string. The method returns [IRefDoc](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetContract Method

Detailed information on the according agreement: agreement number, agreement date including date format template string.

The method returns [IRefDoc](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetAdditionalDoc Method

The invoice-based documents can be used both for the reference of a document number, which should be known to the recipient, as well as for an external document (referenced by a URL) or an embedded document (such as a timesheet as a PDF file). The option of linking to an external document is e.g. required when it comes to large attachments and / or sensitive information, e.g. with personal services, which must be separated from the invoice.

Detailed information on the additional document: document identification or number, URL for the place of external document, type and name of the document, information about the attached binary object, date and time of the document.

This part is restricted for BASIC and CONFORM profiles for ZUGFeRD 1.0. The method returns [IRefDoc](#)

interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

### i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be greater than or equal to 0 and less than number of objects. If c parameter is true, then index parameter can be equal to number of objects for creating of new element.

## GetProcuringProject Method

Detailed information on the associated project: project reference (number, etc.), project name. This part is restricted for ZUGFeRD 1.0. The method returns [IProcProject](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetGrossProductPrice Method

Detailed information on the gross price of the product: gross price, basic and actual amounts and measure units, Detailed information on the discounts and extra charges incl. the reasons for them. This part is allowed within of line item agreements only. The method returns [IPrice](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetNetProductPrice Method

Detailed information on the net price of the product: net price, basic and actual amounts and measure units . Detailed information on the taxation for net price: Included trade tax for B2C, exemption reason if any, appropriated codes. This part is allowed within of line item agreements only. The method returns [IPrice](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetCustomerOrder Method

Detailed information on the order referenced document. That is the order number and the order date with a format template string. This part is allowed within extended profiles only.

The method returns [IRefDoc](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

### i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be greater than or equal to 0 and less than number of objects. If c parameter is true, then index parameter can be equal to number of objects for creating of new element.

## IDelivery Interface

This interface represents the delivery conditions:

- Detailed information on delivery: billed quantity, packaged quantity, free quantity. Unit codes for measurements
- Detailed information on another recipient and final recipient
- Detailed information on delivery documents (delivery bill, advice, delivery notification, etc.)

## GetBilledQuantity Method

Information about billed quantity. The amount of individual items (goods or services) invoiced in the relevant row. The unit code defines the measure unit for the quantity.

The method returns [IQuantity](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetChargeFreeQuantity Method

Information about free of charge goods or services quantity. The unit code defines the measure unit for the quantity.

The method returns [IQuantity](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetPackageQuantity Method

Information about the amount of individual goods packages invoiced in the relevant row. The unit code defines the measure unit for the quantity.

The method returns [IQuantity](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetChainConsignment Method

Detailed information on consignment or shipment. Should be used in case of different goods or services recipient as the buyer. Includes the shipment method, recipient name, identification, details to legal organization, company's register number and its type, trading business name, detailed information to trade contact (person name, department and contact data), detailed tax registration information.

The method returns [IChainConsignment](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetShipToParty Method

Detailed information on the goods or services recipient: ID and global ID, name, description as a legal information, details to the organization, trading business name, trade contact (person name, department and contact data), tax registration.

The method returns [IParty](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetUltimateShipToParty Method3

Detailed information on the final goods or services recipient: ID and global ID, name, description as a legal information, details to the organization, trading business name, trade contact (person name, department and contact data), tax registration.

The method returns [IParty](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetShipFromParty Method

Detailed information on the goods shipper: ID and global ID, name, details to specified legal organization (ID and trading business name), trade contact data (person name, department and contact data), specified tax registration.

The method returns [IParty](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetDeliveryChainEvent Method

Information about the actual delivery time: date of delivery and achievement in terms of taxability.

The method returns [IChainEvent](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetDespatchAdvice Method

Detailed information on the corresponding despatch advice: reference and advice date.

The method returns [IRefDoc](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetReceivingAdvice Method

Detailed information on the associated goods receipt: reference and receipt date.

The method returns [IRefDoc](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetDeliveryNote Method

Detailed information on the corresponding delivery: reference and delivery note date.

The method returns [IRefDoc](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## ISettlement Interface

Detailed information on the payment and billing:

- Bank assigned creditor identifier (Creditor-ID number for SEPA)
- Remittance information
- VAT accounting and invoice currency codes
- Seller reference number: given seller reference number for routing purposes after bilateral agreement
- Detailed information on a business partner: ID and global ID, name, description of a legal organisation, details to a trade contact, tax registration

## PaymentReference Property

Type: [sx\\_str](#). Access: full.

Information about a payment: a textual value used to establish a link between the payment and the invoice, issued by the seller. Helps the seller to assign an incoming payment to the relevant payment process.

## Currency Property

Type: **CurrCode3**. Access: full.

Invoice currency code: the currency in which all invoice amounts are given, except for the total VAT amount in accounting currency. Only one currency shall be used in the invoice, except for the invoice total VAT amount in accounting currency.



## CreditorReference Property

Type: [sx\\_str](#). Access: full.

Bank assigned creditor identifier (Creditor-ID number for SEPA). Unique banking reference identifier of the payee or seller assigned by the payee or seller bank.

## TaxCurrency Property

Type: **CurrCode3**. Access: full.

VAT accounting currency code.

## IssuerReference Property

Type: [sx\\_str](#). Access: full.

Seller reference number. Given seller reference number for routing purposes after bilateral agreement.

## GetInvoiceeParty Method

Detailed information on the invoicee party: ID and global ID, name, specified legal information, details to the organization, trading business name, trade contact (person name, department and contact data), tax registration.

The method returns [IParty](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetPayeeParty Method

Detailed information on the payee party: ID and global ID, name, specified legal information, details to the organization, trading business name, trade contact (person name, department and contact data), tax registration.

The method returns [IParty](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If

parameter is true and element does not exist yet, it will be created and returned.

---

## GetPaymentMeans Method

Payment detailed information and instructions: payment means type code and text, payment card information, account information for payer and payee, information about payment service provider.

The method returns [IPaymentMeans](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetTax Method

Detailed information on a trade tax: VAT breakdown, VAT category tax amount, VAT type code, VAT exemption reason (free text), taxable amount, VAT category code and rate in percent, tax due date.

The method returns [ITax](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetBillingPeriod Method

Detailed information on the invoicing period: description, Invoicing period start and end dates.

The method returns [ITimePeriod](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetAllowanceCharge Method

Detailed information on allowances and charges on the document level: charge indicator, calculation sequence, allowances and charges percentage, base amount and base quantity, actual amount, allowances and charges reason and reason code, VAT type and category codes, allowances and charges VAT rate.

The method returns [IAllowanceCharge](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetLogisticsCharge Method

Detailed information on logistics service fees: description, amount, detailed information on applied trade tax (type, category, rate).

The method returns [ILogisticsCharge](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetPaymentTerms Method

Detailed information about penalties: maturity reference date, due date period, payment penalty base amount, percentage, amount.

The method returns [IPaymentTerms](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetAccount Method

The method returns [IAccount](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetMonetarySummation Method

Detailed information on the document totals: total net amount, charges and allowances on document level, total VAT amount, rounding amount (the amount to be added to the invoice total to round the amount to be paid), gross amount, already paid amount, due for payment and appropriated currency codes.

The method returns [IMonetarySummation](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetReferencedDoc Method

Preceding invoice reference (details on related invoice). A group of business terms providing information about the preceding invoices: the invoice number and date with a format template string.

Information for an additional document can include: document identification or number, URL for the place of external document, type and name of the document, information about the attached binary object, date and time of the document.

To be used in case:

- A preceding invoice is corrected
- Preceding partial invoices are referred to from a final invoice
- Preceding pre-payment invoices are referred to from a final invoice.

The method returns [IRefDoc](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetReceivableAccount Method

Detailed information about the accounting reference: buyer accounting reference, accounting reference type (code).

The method returns [IAccount](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetAdvancePayment Method

Information about the advance payment: paid amount and payment date, detailed information on the included trade tax.

The method returns [IAdvancePayment](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetCurrencyExchange Method

Specification of the invoice currency, local currency and exchange rate for a certain exchange date.

The method returns [ICurrencyExchange](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## ILineItem Interface

Detailed information on an invoice item:

- Item number and codes
- Free text to the item
- Product information and classification
- Country of origin
- Information about the included products

## GetDocument Method

Grouping of general position information: Identifier of the invoice line item, type of the invoice line item (code), subtype of the invoice line item, detailed information about the free text of the line item.

The method returns [ILineDoc](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetProduct Method

An aggregation of business terms, which contains information about the invoiced products and services: global ID, an identification of the item assigned by the seller and the buyer, an article's name and description, item attribute values, detailed information on product classification.

The method returns [ILineProduct](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If

parameter is true and element does not exist yet, it will be created and returned.

---

## GetAgreement Method

Detailed information on the price, order details (order number, reference to the order line item, date.

Detailed information on referenced agreement (number, position, date).

Detailed information on additional referenced document (number, type code, email, binary attachment, date).

The method returns [ILineAgreement](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetDelivery Method

Detailed information on delivery: billed quantity, packaged quantity, free quantity. Unit codes for measurements, deviating recipient and final recipient, delivery documents (delivery bill, advice, delivery notification, etc.).

The method returns [ILineDelivery](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetSettlement Method

Detailed information on the payment and billing at line level: applicable trade tax, invoice line billing period, allowances and charges on line level, information on the item totals, additional referenced document, detailed information on the accounting reference.

The method returns [ILineSettlement](#) interface.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## Inote Interface

Detailed information as a code and a free text to a whole document or to its position. Bilateral agreed text items in form of a code or a free text.

### ContentCode Property

Type: [sx\\_str](#). Access: full.

Bilaterally agreed free text which here is being transmitted as a code.

### Content Property

Type: [sx\\_str](#). Access: full.

A free text containing unstructured information, which is relevant for the positioning of invoice items.

### SubjectCode Property

Type: [sx\\_str](#). Access: full.

Free text of the line item.

## ITimePeriod Interface

Detailed information on the invoicing period: description, Invoicing period start and end dates.

### Description Property

Type: [sx\\_str](#). Access: full.

Invoicing period description (free text).



---

## GetStart Method

Start date of the period.

The method returns [IDateTime](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetEnd Method

End date of the period.

The method returns [IDateTime](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetComplete Method

This method represents the date and time values for a period and format template string according to the regional used formats.

The method returns [IDateTime](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## IDateTime Interface

This interface represents the date and format of this date.

---

## Date Property

Type: [sx\\_time](#). Access: readonly.

The formatted date.

## Format Property

Type: **datetime\_format**. Access: readonly.

The possible date formats are: `datetime_format_CCYYMMDD`,  
`datetime_format_CCYYMM`, `datetime_format_CCYYWW`.

## SetDate Method

The method changes date value.

### f Parameter

Type: **datetime\_format**.

The possible values see above.

### tm Parameter

Type: [sx\\_time](#).

New date value.

## IParty Interface

Detailed information on the trade party: ID and global ID, name, description as a legal information, details to the organization, trading business name, trade contact (person name, department and contact data), tax registration.

## ID Property

Type: [sx\\_str](#). Access: full.

Identification of the trade party.

## Name Property

Type: [sx\\_str](#). Access: full.

Name of the party, usually company name.

## Description Property

Type: [sx\\_str](#). Access: full.

Various legal description of the party.

---

## GetGlobalID Method

Global identification of the party (GLN, DUNS, BIC,ODETTE, ...). The method returns [IGlobalID](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

### i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be greater than or equal to 0 and less than number of objects. If c parameter is true, then index parameter can be equal to number of objects for creating of new element.

---

## GetOrganization Method

Details about the party organization.

The method returns [IOrganization](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

---

## GetContact Method

Detailed information about the trade party contact: contact person name, department name, phone, fax, email contact data and postal address.

The method returns [IContact](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetAddress Method

Contact postal address: post code, address in three lines, city, country, country subdivision.

The method returns [IAddress](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetUniversalURI Method

Details about the contact electronic address.

The method returns [ICommCont](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetTaxID Method

Tax ID or value added tax ID.

The method returns [ITaxID](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

### i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be greater than or equal to 0 and less than number of objects. If c parameter is true, then index parameter can be equal to number of objects for creating of new element.

## IDeliveryTerms Interface

Detailed information on the delivery terms in form of type code.

## IRefDoc Interface

Detailed information on the order referenced document. That is the document number and the document date with a format template string.

Information for an additional document can include: document identification or number, URL for the place of external document, type and name of the document, information about the attached binary object, date and time of the document.

## IssuerAssignedID Property

Type: [sx\\_str](#). Access: full.

Identifier of the associated order, e.g. agreement or order number.

## URIID Property

Type: [sx\\_str](#). Access: full.

Email or another electronic address.

## LineID Property

Identifier of the certain document line.

Type: [sx\\_uint](#). Access: full.

Identifier of the certain document line.

## TypeCode Property

Type: [sx\\_uint](#). Access: full.

The code for identification of a document type.

## RefTypeCode Property

Type: [sx\\_str](#). Access: full.

The code for identification of a referenced document type.

## GetName Method

Document name (free text).

The method returns [sx\\_str](#).

### i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be greater than or equal to 0 and less than number of objects.

## SetName Method

Document name (free text).

The method sets the name.

### c Parameter

Type: [sx\\_str](#).

New name value.

### i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be greater than or equal to 0 and less or equal than number of objects.

## GetAttachment Method

Description of the attachment: binary object, embedded or being sent together with the document. Include MIME code and file name of the object.

The method returns [IAttachment](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetIssueTime Method

Formatting time string including a format string.

The method returns [IDateTime](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## IProcProject Interface

Detailed information on the associated project: project reference (number, etc.), project name.

## IPrice Interface

Detailed information on the gross and net prices of the product: gross price, basic and actual amounts and measure units, detailed information on the discounts and extra charges incl. the reasons for them. Detailed information on the taxation for net price: Included trade tax for B2C, exemption reason if any, appropriated codes.

## IQuantity Interface

Information about billed, free, packaged quantity. The amount of individual items (goods or services) invoiced in the relevant row. The unit code defines the measure unit for the quantity.

## Unit Property

Type: [sx\\_str](#). Access: full.

Unit code for item quantity. Most usable codes: LTR = litre, MTQ = cubic meter, KGM = kilogram, MTR = metre, PCE = piece, TNE = tone.

## Quantity Property

Type: [sx\\_double](#). Access: full.

The quantity of goods items.

## IAmount Interface

The goods value using in differen invoice parts: basic amount, tax amount, actual amount, allowance and charge amount, total amount.

### **Important!**

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

## Currency Property

Type: **CurrCode3**. Access: full.

Currency code according to ISO 4217.

## Amount Property

Type: [sx\\_double](#). Access: full.

Basic amount, tax amount, actual amount, allowance and charge amount, total amount, etc. in accounting currency.

### Important!

When you use trial license, this property can be randomly multiplied by a value from 2 to 9.

## IGlobalID Interface

Global identification of the party (GLN, DUNS, BIC,ODETTE, ...).

## ID Property

Type: [sx\\_str](#). Access: readonly.

Identification of a party or product.

## Format Property

The type of a global indentification.

Type: **global\_id**. Access: readonly.

## SetID Method

Identification of a party or product.

### f Parameter

Type: **global\_id**.

### id Parameter

Type: [sx\\_str](#).



## IOrganization Interface

Details about the organization: ID, name, postal address.

### Name Property

Type: [sx\\_str](#). Access: full.

Trading business name of the party.

### GetAddress Method

Contact postal address: post code, address in three lines, city, country, country subdivision.

The method returns [IAddress](#) interface.

#### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

### GetID Method

Party identification - company registration number.

The method returns [IOrgID](#) interface.

#### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## IContact Interface

Detailed information about the trade party contact: contact person name, department name, phone, fax, email contact data and postal address.

## Person Property

Type: [sx\\_str](#). Access: full.

Person name.

## Department Property

Type: [sx\\_str](#). Access: full.

Department name.

## GetTelephone Method

Detailed information about contact phone number.

The method returns [ICommCont](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetFax Method

Detailed information about contact fax number.

The method returns [ICommCont](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetEmail Method

Detailed information about the contact email address.

The method returns [ICommCont](#) interface.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

# IAddress Interface

Detailed information about the address of the business: post code, address in up to three lines, city name, country ID, country subdivision name.

## Postcode Property

Type: [sx\\_uint](#). Access: full.

Post code of the contact.

## Line1 Property

Type: [sx\\_str](#). Access: full.

Address line 1 of the contact.

## Line2 Property

Type: [sx\\_str](#). Access: full.

Address line 2 of the contact.

## Line3 Property

Type: [sx\\_str](#). Access: full.

Address line 3 of the contact.

## City Property

Type: [sx\\_str](#). Access: full.

City of the contact address.

## Country Property

Type: **CountryCode2**. Access: full.

Country ID of the contact address.

## GetCountryPart Method

Country ID of the contact address.

The method returns [sx\\_str](#).

## i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be greater than or equal to 0 and less than number of objects.

---

## SetCountryPart Method

Country ID of the contact address.

## c Parameter

Type: [sx\\_str](#).

## i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be greater than or equal to 0 and less or equal than number of objects.

---

## ICommCont Interface

---

### Number Property

Type: [sx\\_str](#). Access: full.

Phone or tax number of the contact.

---

### URI Property

Type: [sx\\_str](#). Access: full.

Email address of the contact.

---

### Format Property

Type: [sx\\_str](#). Access: full.

Format string of the number or email address.

---

## IOrgID Interface

Details about the organisation.

## ID Property

Type: [sx\\_str](#). Access: readonly.

Identification of the legal registration of the organization.

## Format Property

Type: **org\_id**. Access: readonly.

Scheme of the identification.

## SetID Method

Identification of the legal registration of the organization.

### f Parameter

Type: **org\_id**.

### id Parameter

Type: [sx\\_str](#).

## ITaxID Interface

Detailed tax information of the trade party.

## ID Property

Type: [sx\\_str](#). Access: readonly.

Tax or VAT number of the party.

## Format Property

Type: **global\_id**. Access: readonly.

Type of the tax ID.

## SetID Method

Tax or VAT number of the party.

### f Parameter

Type: **global\_id**.

## id Parameter

Type: [sx\\_str](#).

## Attachment Interface

Detailed information about the attached binary object: embedded document or a binary object being sent together with the document.

### MimeType Property

Type: [sx\\_str](#). Access: full.

MIME code of the attached object.

### FileName Property

Type: [sx\\_str](#). Access: full.

File name of the attached object.

## ChainConsignment Interface

Detailed information on consignment or shipment. Should be used in case of different goods or services recipient as the buyer.

The information includes:

- the shipment method
- recipient name and identification
- details to legal organization, company's register number and its type, trading business name
- detailed information to trade contact (person name, department and contact data)
- detailed tax registration information.

## ChainEvent Interface

Information about the actual delivery time: date of delivery and achievement in terms of taxability.

## IPaymentMeans Interface

Payment detailed information and instructions:

- Payment means type code and text
- Payment card information: card number, card primary account number, card holder name
- Account information for payer and payee: account identifier, bank information, account name, national account number
- Information about payment service provider (bank information).

## ITax Interface

Detailed information on the trade tax:

- VAT: breakdown, category tax amount, type code, exemption reason (free text), VAT category code and rate in percent
- Goods value of the tax rate, taxable amount, total amount of charges
- Tax due date: value and code.

### TypeCode Property

Type: **tax\_code**. Access: full.

Tax type code. Can be VAT, insurance tax, mineral oil tax, etc. tax.

### ExemptionReason Property

Type: [sx\\_str](#). Access: full.

Exemption reason description for VAT or other tax types.

### CategoryCode Property

Type: **tax\_cat**. Access: full.

VAT category code: coded indication of a sales tax category.

### ExemptionReasonCode Property

Type: [sx\\_str](#). Access: full.

VAT exemption reason code: a coded statement of the reason why the amount is exempted from VAT.

## DateCode Property

Type: **date\_code**. Access: full.

Tax due date, code.

## ApplicablePercent Property

Type: [sx\\_double](#). Access: full.

VAT category rate: the VAT rate represented as percentage that applies for the relevant VAT category.

## GetCalculatedAmount Method

Calculated tax amount on the base of base amount, tax type, percentage.

The method returns [IAmount](#) interface.

### Important!

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetBasisAmount Method

VAT category taxable amount.

The method returns [IAmount](#) interface.

### Important!

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetLineTotalBasisAmount Method

Goods value of the tax rate.

The method returns [IAmount](#) interface.



**Important!**

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

**c Parameter**

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetAllowanceChargeBasisAmount Method

Total amount of charges / allowances.

The method returns [IAmount](#) interface.

**Important!**

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

**c Parameter**

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetTaxPoint Method

Tax due formatted date.

The method returns [IDateTime](#) interface.

**c Parameter**

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## IAllowanceCharge Interface

Detailed information on allowances and charges on the document level: charge indicator, calculation sequence, allowances and charges percentage, base amount and base quantity, actual amount, allowances and charges reason and reason code, VAT type and category codes, allowances and charges VAT rate.

## ILogisticsCharge Interface

Detailed information on logistics service fees: description, amount, detailed information on applied trade tax (type, category, rate).

## IPaymentTerms Interface

Detailed information about penalties: maturity reference date, due date period, payment penalty base amount, percentage, amount.

## IAccount Interface

Detailed information about the accounting reference: buyer accounting reference, accounting reference type (code).

## IMonetarySummation Interface

Detailed information on the document totals:

- Total net, gross and VAT amounts with appropriated currency codes
- Charges and allowances on document level
- Rounding amount (the amount to be added to the invoice total to round the amount to be paid)
- Already paid amount and amount due for payment

## GetLineTotal Method

Sum of invoice line total base or actual amount.

The method returns [IAmount](#) interface.

### **Important!**

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If

parameter is true and element does not exist yet, it will be created and returned.

## GetChargeTotal Method

Sum of invoice line total charge amount.

The method returns [IAmount](#) interface.

### Important!

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetAllowanceTotal Method

Sum of invoice line total allowance amount.

The method returns [IAmount](#) interface.

### Important!

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetTaxBasisTotal Method

Invoice total amount without VAT.

The method returns [IAmount](#) interface.

### Important!

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If

parameter is true and element does not exist yet, it will be created and returned.

## i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be 0 or 1. If c parameter is true, then index parameter can be equal to number of objects for creating of new element.

---

## GetTaxTotal Method

Invoice total VAT amount in accounting currency.

The method returns [IAmount](#) interface.

### **Important!**

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be 0 or 1. If c parameter is true, then index parameter can be equal to number of objects for creating of new element.

---

## GetRounding Method

Rounding amount: the amount to be added to the invoice total to round the amount to be paid.

The method returns [IAmount](#) interface.

### **Important!**

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetGrandTotal Method

Invoice total amount with VAT (Gross amount).

The method returns [IAmount](#) interface.

### Important!

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

### i Parameter

Type: [sx\\_uint](#).

The index of an element. Can be greater than or equal to 0 and less than number of objects. If c parameter is true, then index parameter can be equal to number of objects for creating of new element.

## GetTotalPrepaid Method

Already paid amount: sum of amount paid in advance.

The method returns [IAmount](#) interface.

### Important!

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

### c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetTotalAllowanceCharge Method

Total amount of allowances and charges.

The method returns [IAmount](#) interface.

### Important!

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## GetDuePayable Method

Amount due for payment.

The method returns [IAmount](#) interface.

### **Important!**

When you use trial license, Amount property can be randomly multiplied by a value from 2 to 9.

## c Parameter

Type: [sx\\_bool](#).

If parameter is false, the method returns existing element only and null if element does not exist yet. If parameter is true and element does not exist yet, it will be created and returned.

## IAvancePayment Interface

Information about the advance payment: paid amount and payment date, detailed information on the included trade tax.

## ICurrencyExchange Interface

Specification of the invoice currency, local currency and exchange rate for a certain exchange date.

## ILineDoc Interface

Grouping of general position information: Identifier of the invoice line item, type of the invoice line item (code), subtype of the invoice line item, detailed information about the free text of the line item.

## ILineProduct Interface

An aggregation of business terms, which contains information about the invoiced products and services: global ID, an identification of the item assigned by the seller and the buyer, an article's name and description, item attribute values, detailed information on product classification.

## ILineAgreement Interface

Detailed information on the agreement at line level, it is mostly similar to the [IAgreement](#) interface.

## ILineDelivery Interface

Detailed information on delivery at line level, it is mostly similar to the [IDelivery](#) interface.

## ILineSettlement Interface

Detailed information on the payment and billing at line level, it is mostly similar to the [ISettlement](#) interface.

## Enumerations

### einv\_options

The enumeration defines options for opening or creating of invoices.

#### einv\_options\_def

For invoice creating this option declares ZUGFeRD 1.0 standard, please note that this standard is outdated.

#### einv\_options\_zugferd20

For invoice creating this option declares ZUGFeRD 2.0 standard, please note that this standard is outdated.

#### einv\_options\_zugferd21

For invoice creating only, this option declares ZUGFeRD 2.1.x standard.

#### einv\_options\_facturx

For invoice creating only, this option declares FacturX standard.

#### einv\_options\_xrechnung

For invoice creating only, this option declares ZUGFeRD 2.1.x standard with embedded XRechnung 1.2.2

invoice.

### **einv\_options\_xrechnung20**

For invoice creating only, this option declares ZUGFeRD 2.1.x standard with embedded XRechnung 2.0 invoice.

### **einv\_options\_pdfa\_3b**

For invoice creating only: create PDF/A-3b invoice instead of PDF/A-3a.

### **einv\_options\_check\_xml**

For invoice opening only, this option specifies that Open method must load embedded XML invoice for simple checking: required namespaces, root node, profile definition, correspondence between invoice standard (and profile) declared in PDF/A-3 and XML invoices.

### **einv\_options\_check\_pdfa**

For invoice opening only, this option specifies that Open method must validate complete PDF/A conformity of PDF/A invoice.

---

## **einv\_status**

The enumeration defines validation status for opening of invoice file.

---

## **einv\_standard**

The enumeration defines supported standards of invoices.

---

## **einv\_profile**

The enumeration defines profiles of invoices.



# TECHNICAL SUPPORT

E-Mail: [pdf@soft-xpansion.com](mailto:pdf@soft-xpansion.com)