softxpansion

PDF Xpansion SDK 18 Reference

soft Xpansion GmbH & Co. KG • https://soft-xpansion.de

TABLE OF CONTENTS

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#, VB.NET, Delphi, VBA and etc. The intended audience includes but is not limited to these languages. Familiarity with the PDF, e-invoice and e-order formats and the object models can be helpful.

Different development platforms in this Reference have been designated as:

- **C++** – applications and services written in C++ using Microsoft Visual Studio
- **.NET** – .NET applications written in C# and VB.NET
- **OLE** – applications written in the languages which support **OLE Automation Protocol** -
  MS Office VBA, products of Embarcadero platform (for example Delphi, C++ Builder and etc.)

System Requirements

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

- Windows 11
- Windows Server 2022
- Windows 10
- Windows Server 2016

C++

All SDK functionality for this platform is implemented in **pdf-xpansion-windesktop-x64.dll** / **pdf-xpansion-windesktop-x86.dll** libraries and does not requires any additional components.

.NET

All SDK functionality for this platform is provided by assemblies built in **Visual Studio 2026** for
.NET 10.0 / 9.0 / 8.0
.NET Framework 4.8

Important! Any SDK assemblies are wrappers for unmanaged libraries and require **pdf-xpansion-windesktop-x64.dll** / **pdf-xpansion-windesktop-x86.dll** libraries to be available during use.
Thus, they cannot be used outside the Windows platform.

The **.NET Framework SDK assemblies** require the redistributable files for **Visual Studio 2026** and appropriate version of **.NET Framework** to be installed.

OLE

All SDK functionality for this platform is implemented using the OLE Automation technology in **pdf-xpansion-com-x64.dll** / **pdf-xpansion-com-x86.dll** libraries.
For MS Office VBA projects we recommend to use **pdf-xpansion-vba-x64.dll** / **pdf-xpansion-vba-x86.dll** libraries because VBA supports not all base types supported by OLE.
Important! The SDK OLE libraries are wrappers and require **pdf-xpansion-windesktop-x64.dll** / **pdf-xpansion-windesktop-x86.dll** libraries to be available during use.

Installation of SDK

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

.NET

Important! The **.NET Framework assemblies** require the redistributable files for **Visual Studio 2026** and appropriate version of **.NET Framework** to be installed.

OLE

If you want to create main SDK object (get entry point of SDK library) using **CreateInstance** / **CreateObject** methods, then SDK OLE library must be registered using **regsvr32** utility. The SDK OLE library can be used without registration, if you can call a C-compatible function instead of using CreateInstance / CreateObject methods within your projects (see detailed information here).

Contents of SDK package

Folder “Docs”

The folder contains SDK documentation.

Folder “DevRes”

The folder contains developer resources and SDK samples.
These files cannot be redistributed with your application.

Subfolder “include”

C++:
The folder contains header files for classic C++ applications. You don’t need these files for other developer 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”.

Reference SDK libraries in Your Projects

C++

Folder "**DevRes\include"** contains the header files for C++ desktop or server applications. You should include these files (all or used part) 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.

.NET

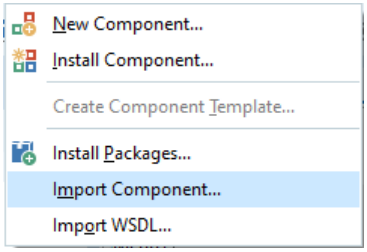
Add a reference to the appropriate assembly from **redist** folder of the **PDF Xpansion SDK** to your project. All assemblies (except .NET Framework assemblies) are compiled for **AnyCPU** target. If you need **PDF Viewer** functionality, you need a reference for additional SDK assembly also.

Important! The **.NET Framework assemblies** require the redistributable files for **Visual Studio 2026** and appropriate version of **.NET Framework** to be installed. These assemblies are compiled for **x86** and **x64** targets.

OLE

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).

Register **pdf-xpansion-vba-x86.dll** or **pdf-xpansion-vba-x64.dll** using **regsvr32** utility for VBA projects.



Embarcadero developers must import SDK OLE library in the project.
Use "**Component\Import Component**" function of Embarcadero environment.
This function lets you import the types, objects, and interfaces from SDK OLE library and can optionally generate component wrappers for all creatable CoClasses, SDK interfaces, structures and enumerations in the language used in your project. It creates a file with unit/namespace **PDF_Xpansion_SDK_18_TLB**.
This file contains declarations for the classes, types and interfaces defined in the SDK OLE library.
By including it in your project, those definitions are available to your application so that you can create objects and call their interfaces.

Replace Trial License

After you purchase the PDF Xpansion SDK you get your corporate license (file **pdf-xpansion.license**) 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 file of your personal license 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.

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).

After this, 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.

C++

You must copy all header files from folder "**DevRes\include"** to all your projects where you have updated SDK libraries because some releases extend SDK interfaces. All these projects should be recompiled.
Important! Compilation of your projects with headers from previous SDK releases or starting your applications with updated SDK libraries without recompiling will result in hard-to-detect crashes.

OLE

You must repeat COM registration and import it as a component, if you use it in the projects at Embarcadero platform.

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 applications 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.

.NET

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**.
Important! The **.NET Framework assemblies** require the redistributable files for **Visual Studio 2026** and appropriate version of **.NET Framework** to be installed.

OLE

You must register all redistributed OLE/VBA libraries on the client PCs before use,
if you use CreateInstance / CreateObject method for creating of main SDK object.
The SDK OLE libraries can be used without registration, if you use a C-compatible function instead of CreateInstance / CreateObject method within your projects (see detailed information here).

List of redistributable files

These files must be redistributed with any application which uses PDF Xpansion SDK, including applications which use .NET or OLE/VBA libraries of SDK**.**

pdf-xpansion-windesktop-x86.dll **pdf-xpansion-windesktop-x64.dll**	The main library of PDF Xpansion SDK. If you build only x86 or x64 binaries in your project, you may redistribute accordingly x86 or 64 libraries only.
pdf-xpansion.license	The license file of PDF Xpansion SDK. **Important!** After you have licensed the PDF Xpansion SDK, you will get your corporate license file and you must redistribute your corporate license only. The license file placed in SDK package is trial license, **redistribution with a trial license is not permitted**.
pdf-xpansion.pds	The resources store of PDF Xpansion SDK.
eInvoice-template.invtpl	The design template for e-invoice visualization – can be optionally redistributed for the transaformation of XML e-invoices to the human readable form (PDF files).

These files must be redistributed with .NET application which uses PDF Xpansion SDK. Please note, you need to redistribute assemblies for used .NET Framework version and processor architecture only. This .NET Framework version must be available at the target workplace. You may place these assemblies in Global Assembly Cache (GAC) also, but you must use EntryPointEx instead of EntryPoint for SDK initialisation in this case.

pdf-xpansion-net80.dll pdf-xpansion-net90.dll pdf-xpansion-net100.dll	The .NET assemblies of PDF Xpansion SDK. They are compiled for AnyCPU target.
pdf-xpansion-net80-forms.dll pdf-xpansion-net90-forms.dll pdf-xpansion-net100-forms.dll	The assemblies with the document viewer as a Forms Control. They are compiled for AnyCPU target.
pdf-xpansion-net80-wpf.dll pdf-xpansion-net90-wpf.dll pdf-xpansion-net100-wpf.dll	The assemblies with the document viewer as a WPF Control. They are compiled for AnyCPU target.
pdf-xpansion-net48-x86.dll pdf-xpansion-net48-x64.dll	The .NET Framework 4.8 assemblies of PDF Xpansion SDK. They are compiled for x86 and x64 targets.
pdf-xpansion-wpf48-x86.dll pdf-xpansion-wpf48-x64.dll	The assemblies with the document viewer as a WPF Control. They are compiled for x86 and x64 targets, they use .NET Framework 4.8 assemblies of appropriate version.

These files must be redistributed with applications written in the languages which support **OLE Automation** Protocol - MS Office VBA, products of Embarcadero platform (for example Delphi, C++ Builder and etc.)

pdf-xpansion-com-x86.dll pdf-xpansion-com-x64.dll	The SDK OLE libraries.
pdf-xpansion-vba-x86.dll pdf-xpansion-vba-x64.dll	The SDK OLE libraries for MS Office VBA projects.

These files can be optionally redistributed with application which uses localizable resources, f.e. PDF stamps**.**

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.
5. Please read the chapter **Using Document Viewer** if you want integrate document viewer in your application.
6. Please read the chapter **Electronic invoice standards and legal basis** if you want process e-invoices and/or orders.
7. 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.
8. Please read the chapter **Error Handling / Exceptions** about processing of SDK errors in your projects.
9. 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 PDF Xpansion 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.

C++

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** functions. After successful loading of library, you can use module handle to get entry point of SDK library anytime. See our sample as an example.

OLE

Optionally you can use the library without registering at the target workplace. 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** C-compatible functions in these libraries, if your environment allows this operation (for example, Embarcadero projects). After successful loading of library, you can use module handle to get entry point of SDK library anytime.

Entry Point of SDK Library

All the functionality of the SDK is represented by a number of interfaces, which can be accessed sequentially from the root (main) interface, which is IApplication. We recommend you to get this interface 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, immediately after getting it.

C++

At first, you should get address of **EntryPoint** function using **GetProcAddress** Windows function. Than you can cast a retrieved address to **SX::LibraryEntryPoint** and call **EntryPoint** function.

.NET

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 should get IApplication and authorize it before creating of any these controls.
If you place the PDF Xpansion SDK assemblies in Global Assembly Cache (GAC), you must use **EntryPointEx** method of the SX.NET.PDFXpansionSDK class, instead of EntryPoint method.

OLE

you should create an object of PDFXpansionSDK class (ID is "**SX.PDFXpansionSDK18**") and use **EntryPoint** or **EntryPointEx** method of this object to get OLE interface IApplication.

Optionally you can use the SDK OLE library (pdf-xpansion-com-x86.dll or pdf-xpansion-com-x64.dll) without registration in registry - you should get address of **EntryPoint** or **EntryPointEx** function using **GetProcAddress** Windows API function. Than you can call this function which returns OLE interface IApplication also (see our Embarcadero samples for this technique).

If you use **ActiveX** in your project, you should get IApplication and authorize it before creating of this control.

EntryPoint Function

This function is available in the next SDK libraries: pdf-xpansion-windesktop-x86.dll, pdf-xpansion-windesktop-x64.dll, pdf-xpansion-com-x86.dll and pdf-xpansion-com-x64.dll.
It hasn't the parameters and returns IApplication interface.

OLE

When you use this function in OLE library, you must have the native library (pdf-xpansion-windesktop-xXX.dll) in the same folder as OLE library and native library should be the same version. Otherwise function returns NULL. You can determine the specific cause of the error by calling the **GetAppStatus** function, it returns HRESULT.
See **EntryPoint** method of PDFXpansionSDK class, as an alternative for this function.

EntryPointEx Function

OLE

This function is available in the SDK OLE libraries only. It has single parameter and returns IApplication interface. Unlike the **EntryPoint** function, this function allows you to use OLE library that is placed separate from the rest of the redistributable SDK files. The parameter (LPCTSTR type) is absolute path (including filename) of native library (both libraries should be the same version).
Function returns NULL if it can't find or load native library. You can determine the specific cause of the error by calling the **GetAppStatus** function, it returns HRESULT.
See **EntryPointEx** method of PDFXpansionSDK class, as an alternative for this function.

GetAppStatus Function

This function exists in SDK OLE libraries only. It hasn't the parameters and returns HRESULT with error code which explains the reason of problem with **EntryPoint** or **EntryPointEx** methods:

- ErrorInvalidFormat – filename of OLE library is changed
- ErrorPathNotFound – native library was not found
- ErrorInvalidUse – native library is wrong
- ErrorOutOfRange – used library and native library have different versions

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.

C++

PDF Xpansion SDK throws C++ exceptions of the **SX::IException** type.

.NET

PDF Xpansion SDK throws exceptions of the **SX.NET.Exception** type (inherits System.Exception).

OLE

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 OLE.

Please note that Embarcadero wrappers (Delphi / C++ Builder) throw EOleSysError exception for any SDK method call with unsuccessful **HRESULT**.

You can use **GetLastErr** method of IApplication interface to get status (HRESULT) of any last called SDK method also.

For example, in VBA for Microsoft Office you can use this GetLastErr to check the result of calling important method, since VBA does not provide **HRESULT** to your code. You need to call it immediately after calling such method.

Files and Streams

Multiple methods and properties of SDK operate with data blocks (streams) which can contain binary, XML or text data. PDF Xpansion SDK provides a universal ISequentialStream interface that is used in all such methods and properties. This interface is similar to the well-known COM interface of the same name, but is not the same. It can only be used within the SDK API, not as a replacement for the ISequentialStream COM interface. Similarly, ISequentialStream COM interface cannot be used in its place.

PDF Xpansion SDK operates both memory-based blocks and files on thebase of this interface.

Raster Images (Bitmaps)

Some SDK methods or properties operate with raster images (bitmas) and PDF Xpansion SDK provides a universal IBitmapData interface that is used in all such methods and properties.

In some cases where bitmaps are used as method parameters, SDK provides alternative methods with the ISequentialStream parameter that assume the content of a JPEG compressed raster image only, such as when inserting into a PDF document, **thus avoiding recompression and additional quality loss**.

Electronic Invoices and Orders

The PDF Xpansion SDK provides complete functionality for processing of electronic invoices and orders. It could be the PDF/A based (ZUGFeRD, Factur-X, Order-X) and XML based (XRechnung, ZUGFeRD, Factur-X, Order-X, Peppol BIS) documents.

Combine, Convert, Import and Export the PDF 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.

Make PDF/A conform files from PDF files

The library makes conform PDF/A (versions 1, 2, 3 and 4) files from PDF files or images.

Create, Edit or Explore the page content of the PDF documents

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.

Document Viewer (PDF, e-invoices, etc.)

The PDF Xpansion SDK provides several platform-oriented implementations of document viewer. Using these implementations, you can easy and quickly integrate power viewer and editor of **PDF**, **XPS** and other supported document formats 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 technology **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.

C++

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 windowless viewer object (**SX::Viewer:IViewer** interface) to manage viewport with layout of document in your own window.

.NET

On this platform you can use viewer as an **Forms Control** or **WPF Control**. These controls are derived from the corresponding objects in both the .NET Framework and pure .NET environments. So, you can integrate it in your applications in the traditional way.

OLE

On this platform you can use viewer as a classic **ActiveX** 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 realtive 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**.

Window-based viewer

PDF Xpansion SDK implements the window-based viewer which inherits from CWindowImpl. It enabling direct window manipulation (HWND management) or using SX::Viewer::IWinViewer interface and processes all necessary window messages.

.NET Forms Control

The .NET Framework assemblies of PDF Xpansion SDK implement the **PDFXpansionFormsControl** themselves. So, you find it in any pdf-xpansion-net48-xXX.dll library.
This control is derived from System::Windows::Forms::Control.

If you use PDF Xpansion SDK assemblies for .NET 8 and higher, you need include in your project a pdf-xpansion-netXX-forms.dll library, where the **PDFXpansionFormsControl** is implemented.
This control is derived from System::Windows::Forms::UserControl.

The control properties, methods and events for different environments are almost identical, please find more detailed information here.

.NET WPF Control

If you use PDF Xpansion SDK assemblies for .NET Framework, you need include in your project a pdf-xpansion-wpf48-xXX.dll library, where the **PDFXpansionWPFControl** is implemented.
If you use PDF Xpansion SDK assemblies for .NET 8 and higher, you need include in your project a pdf-xpansion-netXX-wpf.dll library, where the **PDFXpansionWPFControl** is implemented.
This control is derived from System.Windows.Controls::UserControl.

The control properties, methods and events for different environments are almost identical, please find more detailed information here.

ActiveX Control

The OLE library of PDF Xpansion SDK implements the **PDFXpansionControl** as a classic ActiveX control. You can create a control object using ID "**SX.PDFXpansionControl18**". For compatibility with earlier versions of SDK, a simpler and a slightly outdated version of the control can be used (ID "SX.PDFXpansionViewer18"), but in future versions of SDK it will no longer be there.

The control properties, methods and events for different environments are almost identical, please find more detailed information here.

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.

.NET: For this platform, root namespace is "**SX.NET".**

OLE This platform does not use any namespaces.

IRefObject Interface / IDisposable Interface

C++

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.

.NET:

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 immediatelly.

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)

C++

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 in the .NET and OLE libraries.

All the functionality of the SDK is represented by a number of interfaces, which can be accessed sequentially from the root (main) interface, which is IApplication. The methods of PDFXpansionSDK returns this interface. We recommend you to get this interface 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, immediately after getting it.

.NET

If you place the PDF Xpansion SDK assemblies in Global Assembly Cache (GAC), you must use **EntryPointEx** method instead of EntryPoint method.

OLE

you can create an object of **PDFXpansionSDK** using **CreateInstance** / **CreateObject** functions - ID is "**SX.PDFXpansionSDK18**". The OLE library should be previously registered in registry.

Optionally you can use the SDK OLE library without registration in registry - see description of **EntryPoint** and **EntryPointEx** functions.

EntryPoint Method

Method hasn't the parameters and returns IApplication interface. When you use this method, you must have the native library (pdf-xpansion-windesktop-xXX.dll) in the same folder as used library and native library should be the same version.

OLE

Method returns NULL if it failed. You can determine the specific cause of the error by calling the **GetAppStatus** method, it returns HRESULT.

.NET:
Method throws an exception if it failed. You can determine the specific cause of the problem by error code of exception.
Important! All assemblies compiled for AnyCPU target load **pdf-xpansion-windesktop-x64.dll** native library. They have **EntryPointX86** method also, which loads **pdf-xpansion-windesktop-x86.dll**.

EntryPointEx Method

Method has single parameter and returns IApplication interface. Unlike the EntryPoint method, this method allows you to use library that is pplaced separate from the rest of the redistributable SDK files. The parameter is absolute path (including filename) of native library, for example "C:\Program Files\Company\Product\sx\pdf-xpansion-windesktop-x86.dll". Both libraries should be the same version.

OLE

Method returns NULL if it failed. You can determine the specific cause of the error by calling the **GetAppStatus** method, it returns HRESULT.

.NET:
Method throws an exception **SX.NET.Exception** if it failed. You can determine the specific cause of the problem by error code of exception.

GetAppStatus Method

This method exists in SDK OLE libraries only. Method hasn't the parameters and returns HRESULT with error code which explains the reason of problem with **EntryPoint** or **EntryPointEx** methods.

Errors in EntryPoint / EntryPointEx

The reasons of problem with **EntryPoint** or **EntryPointEx** methods:

- ErrorInvalidFormat – filename of OLE or .NET library is changed
- ErrorPathNotFound – native library was not found
- ErrorInvalidUse – native library is wrong
- ErrorOutOfRange – used library and native library have different versions

SX::IException / SX.NET.Exception

Calling of any methods of PDF Xpansion SDK, getting/setting of properties can throw the exception in the case of troubles in the C++ and .NET libraries. The reason of exception you can get using **ErrorCode** property of exception interface/class. The SDK OLE library uses other mechanism of error notification.

C++

PDF Xpansion SDK throws C++ exceptions of the **SX::IException** type.

.NET

PDF Xpansion SDK throws exceptions of the **SX.NET.Exception** type (inherits System.Exception).

ErrorCode Property

Type: ErrorCode. Access: readonly.

The reason of exception.

IApplication Interface

All the functionality of the SDK is represented by a number of interfaces, which can be accessed sequentially from the root (main) interface, which is IApplication. We recommend you to get this interface 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 **Authorize** method during the initialize phase of your application, immediately after getting it.

Version Property

Type: sx_str. Access: readonly.

The property provides technical version (release) 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 “first level” 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 folder as the native library (pdf-xpansion-windesktop-xXX.dll) or by path specified as a second parameter of method.

Please note, every license file has only one suitable license key, you can't authorize SDK using trial license file and your license key or vice versa – you license file and trial license key.

It's also important to know that a license authorizes only the SDK version (generation) for which it was issued (e.g., 17; the release number is unimportant and can be any). When upgrading from one SDK generation to a new one, you must obtain a license/key pair exactly for that version and replace them in all projects that will use the new version.

Important! Please, don't redistribute license key file with your application.

LicenseKey Parameter

Type: sx_str.

The license key, you get it together with your corporate license, but unlike the license file, you should use license key string in your sources to call this method only.

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

LicenseFilePath Parameter

Type: sx_str.

This is optional parameter - if you redistribute the license file with different filename and/or is not in the same directory as the native library (pdf-xpansion-windesktop-xXX.dll), you should specify absolute path of license file here, otherwise this parameter must be null or empty string.

Errors

The method produces an exception (or failed HRESULT in SDK OLE library):

- ErrorPathNotFound – license file was not found
- ErrorAuthFailed - license key doesn't match the license file or license file is corrupted
- ErrorAccessDenied – main SDK resource file (**pdf-xpansion.pds**) was not found
- ErrorOutOfRange – trial license is expired, please download actual version of PDF Xpansion SDK

GetLastErr Method

This method is available in SDK OLE library only. You can use it to get status (HRESULT) of any last called SDK method if you can't use HRESULT of called method.

For example, in VBA for Microsoft Office you can use this method to check the result of calling important method, since VBA does not provide HRESULT to your code. You need to call it immediately after calling such method.

IAppFactory Interface

The interface represents factory of SDK objects. Please note, that it instantiates the SDK objects permitted by license only.

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.

FilePath Parameter

Type: sx_str.

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

Errors

The method produces an exception ErrorPathNotFound if cannot 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.

FilePath Parameter

Type: sx_str.

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

Errors

The method produces an exception ErrorPathNotFound if cannot find or open the file.

CreateRefStream Method

The method creates new instance of **wrapper stream object** for the platform specific data stream.
It returns ISequentialStream interface.

Stream Parameter

C++

The IStream COM interface. You can use the streams created by CreateStreamOnHGlobal or StgCreateStorageEx Windows functions or any other implementation of **IStream**.

.NET

The System.IO.Stream derived class. You can use any stream implementations available on .NET platform.

OLE

The IStream COM interface. You can use any implementation of native COM **IStream**.

CreateDualStream

The method creates new instance of dual-nature stream - until its size exceeds the specified limit, it will be memory-based. If the limit is exceeded, it will be saved as a temporary file, which will slow down its operation but allow for optimal use of working memory in the case of truly large stream volumes. It returns ISequentialStream interface.

Limit Parameter

Type: sx_uint.

The maximum number of MB to be stored in memory.

CreateBitmapFromStream Method

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

The method produces the exceptions if failed, see ErrorCode property of exception for details.

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 IBitmapData interface.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

.NET: The method isn't available on this platform.

CreateBitmapFromBits Method

The method creates new instance of bitmap object with the specified width, height, pixel format and using array of bytes containing the pixel data for the bitmap. The method returns IBitmapData interface.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

Width Parameter

Type: sx_uint.

Bitmap width in pixels.

Height Parameter

Type: sx_uint.

Bitmap height in pixels.

Format Parameter

Type: PixelFormat.

Bitmap pixel format. Actually any indexed formats are not supported.

pBits Parameter

Type: sx_byte*.

The pointer to the memory buffer - address of bytes array containing the pixel data for the bitmap. The stride of lines padded out to a byte.

.NET

Type of parameter is byte[].

OLE

Type of parameter is const BYTE*.

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.

CreateDrawingContext Method

The method creates new instance of SDK drawing context object. The method returns Graphics::IDrawingContext interface.

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.

CreateEInvoiceDocument Method

The method creates new instance of PDF/A-3 based invoice/order document. The method returns EInvoice:: IInvoiceDocument interface.

CreateViewer Method

The method creates new instance of viewer object. The method returns Viewer::IViewer interface.

C++ We recommend you create window based viewer instead of this object.

.NET: 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.

C++ You must define **SX_WINDESKTOP_DLL** macro in your project if you need to use window based viewer.

.NET: 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.

C++ 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.

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.

.NET:

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 completelly resets the document - all pages and other document ressources 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 warpper 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.

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.

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 (it can be binary, text or XML data of any size).

You can create objects implementing this interface and associated with both disk files or memory blocks using IAppFactory interface: **CreateSequentialStream**, **CreateReadStream**, **CreateWriteStream**, **CreateRefStream**, **CreateDualStream**. SDK methods or properties that return this interface create the corresponding object themselves (usually as a memory block).

C++

The **SX::ISequentialStream** interface is derived from IRefObject.
Optionally you can create object with this interface which is wrapper to IStream COM interface using **IAppFactory::CreateRefStream** method. This way you can use the streams created by CreateStreamOnHGlobal or StgCreateStorageEx Windows functions or any other implementation of **IStream**.

.NET

The **SX.NET.ISequentialStream** is derived from **System.IDisposable** interface. When you finish using the stream object, you need to dispose it and reclaim the used by this object unmanaged resources immediately - file will be closed, allocated memory block will be freed.
Optionally you can create object with this interface which is wrapper to any System.IO.Stream derived class using **CreateRefStream** method of IAppFactory. Thus, you can use any stream implementations available on this platform in SDK.

And finally, the **System.IO.XpansionStream** class, implemented in the PDF Xpansion SDK, inherits the functionality of the System.IO.Stream. When you get ISequentialStream interface from any SDK method or property, you can create XpansionStream from this ISequentialStream and continue working with the stream through the familiar System.IO.Stream interface.

OLE

Important! PDF Xpansion SDK interface for this element is **SX_ISequentialStream**, it's not native COM **ISequentialStream**.
Optionally you can create object with this interface which is wrapper to IStream COM interface using **IAppFactory::CreateRefStream** method. This way you can use any implementation of native COM **IStream**.

CanRead Method

The method returns true if this stream supports reading.

CanRead Method

The method returns true if this stream supports writing.

GetLength Method

The method returns actual length in bytes of the stream.

GetPos Method

The method returns actual position in bytes of the stream.

Reset Method

The method sets the stream pointer to the beginning of the stream data. We recommend calling this method for all streams you receive from SDK methods or proerties.

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.

Buf Parameter

Type: sx_byte*.

The pointer to the memory buffer.

.NET

Type of parameter is byte[].

OLE

Type of parameter is BYTE*.

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.

Buf Parameter

Type: sx_byte*.

The pointer to the memory buffer.

.NET

Type of parameter is byte[].

OLE

Type of parameter is const BYTE*.

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.

C++ This interface is derived from IRefObject at this platform.

ICryptoKey Interface

The interface supports simplified sequential access to stream data.

C++ 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:

SX	Windows	.NET	OLE	VBA
sx_bool	bool	System::Boolean	VARIANT_BOOL	VARIANT_BOOL
sx_uint (sx_flags)	unsigned int	System::UInt32	ULONG	LONG
sx_int	int	System::Int32	LONG	LONG
sx_byte	unsigned char	System::Byte	BYTE	BYTE
sx_short	unsigned short	System::Int16	USHORT	SHORT
sx_qword	unsigned __int64	System::UInt64	ULONGLONG	LONG
sx_float	float	System::Single	double	double
sx_double	double	System::Double	double	double
sx_str (IStr)	const wchar_t*	System::String^	BSTR	BSTR

IStr Interface

C++ 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.

.NET: 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.

.NET: 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.

.NET: 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.

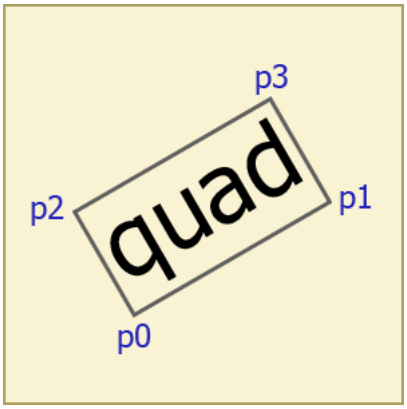
width

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.

.NET: 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

IBitmapData Interface

The interface represents a raster image (bitmap) - set of pixels. The SDK bitmap class which implements this interface supports various pixel formats – color spaces (RGB, CMYK, grayscale, indexed) and bit depth. Addtionally, it supports alpha chanel for RGB pixels or for RGB colors in palette of indexed bitmaps optionally.

The IBitmapData represents single frame images only, but after loading multiframe images from file formats which support this possibility (f.e. TIFF) you can get the chain of bitmap objects (frames) and walk along this chain using **GetNextFrame** method.

PDF Xpansion SDK serializes bitmap objects using Windows Imaging Component. Therefore, you can load SDK bitmap objects from files of any supported formats by IWICBitmapDecoder, and also save bitmaps to files of formats supported by IWICBitmapEncoder.

The PDF format, in turn, allows a variety of formats of embedded raster images, including those that have no equivalent file format. SDK provides all necessary transformations for such images when extracting them from PDF documents or rendering pages containing them. When embedding new raster images into PDF document, the SDK transforms bitmap to the most suitable format for embedding it into PDF document.

You can create bitmap object from image file (as ISequentialStream) using IAppFactory interface **CreateBitmapFromStream**.

C++

The **SX::Graphics::IBitmapData** interface is derived from IRefObject.

Optionally you can create object with this interface from IWICBitmapSource using **IAppFactory::CreateBitmapFromSource** method. Also you can create bitmap object with the specified width, height, pixel format and array of bytes containing the pixel data for the bitmap using **IAppFactory::CreateBitmapFromBits** method. And finally, you can create bitmap object from GDI HBITMAP using **IAppFactory::CreateBitmapFromHandle** method.

.NET

The **SX.NET.Graphics.IBitmapData** is derived from **System.IDisposable** interface. When you finish using the bitmap object, you need to dispose it and reclaim the used by this object unmanaged memory immediately.
Optionally you can create bitmap object with the specified width, height, pixel format and array of bytes containing the pixel data for the bitmap using **IAppFactory::CreateBitmapFromBits** method.

Also the **SX.NET.Drawing.Bitmap** class (located in **pdf-xpansion-netXX-forms.dll** SDK library) provides a static method **Create** which creates SDK bitmap from **System.Drawing.Bitmap** and static method **Copy** for creation of **System.Drawing.Bitmap** from **SX.NET.Graphics.IBitmapData**. The library **pdf-xpansion-netXX-wpf.dll** provides the same class for conversion between **System.Windows.Media.Imaging.BitmapSource** and **SX.NET.Graphics.IBitmapData**.

OLE

The **Graphics_IBitmapData** interface represents SDK bitmap object.

Optionally you can create object with this interface from IWICBitmapSource using **IAppFactory::CreateBitmapFromSource** method.

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 gets/sets DPI of bitmap optionally saved in file. When DPI is undefined, the value is 0. **Please note**, this property doesn't represents/changes the DPI of image embedded to PDF or drawing DPI.

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 another pixel 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 bytes array containing the pixel data for the bitmap. 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.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

.NET: The method retrieves address of unmanaged memory.

GetScanLine Method

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

The method produces the exceptions if failed, see ErrorCode property of exception for details.

.NET: The method retrieves address of unmanaged memory.

Line Parameter

Type: sx_uint.

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

GetNextFrame Method

The method returns next bitmap object in the chain of multiframe image.

CopyAs Method

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

The method produces the exceptions if failed, see ErrorCode property of exception for details.

Format Parameter

Type: PixelFormat.

The preferred pixel format of new bitmap.

CopyToSource Method

Method copies current bitmap as WIC Bitmap object and returns IWICBitmapSource. Please note that **you must release this object** when you no longer need to use it.

SaveToStream Method

The method saves bitmap object to the stream in specified file format.

Stream Parameter

Type: ISequentialStream.

The stream for save. Please reset stream before save.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

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).

Crop Method

The method creates new bitmap as cropped copy of this bitmap in the same pixel format. Current bitmap remains unchanged.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

Rc Parameter

Type: sx_rect.

The left-top coordinates and size of cropping area.

Rotate Method

The method creates new bitmap as rotated copy of this bitmap (90, 180 or 270 degrees) in the same pixel format. Current bitmap remains unchanged.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

R Parameter

Type: sx_rotation_angle.

Rotation value.

Resize Method

The method creates new bitmap as resized copy of this bitmap in the same pixel format. Current bitmap remains unchanged.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

Scale Parameter

Type: sx_double.

Scale value (1.0 – no scale).

Rsw Parameter

Type: sx_uint.

New bitmap width in pixels.

Rsh Parameter

Type: sx_uint.

New bitmap height in pixels.

Interp Parameter

Type: ImageInterpolation.

Scale interpolation method.

Flip Method

The method creates new bitmap as flipped copy of this bitmap in the same pixel format. Current bitmap remains unchanged.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

Hrz Parameter

Type: sx_bool.

Flip direction.

Paste Method

The method changes current bitmap, it replaces pixels in the rectangle area defined by left-top coordinates of the current bitmap and size of source bitmap. Both bitmaps must be in the same pixel format, area of replacement must be completely within size of current bitmap.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

pBmp Parameter

Type: IBitmapData.

The source bitmap.

X,Y Parameter

Type: sx_uint.

The coordinates of replacement area.

Blend Method

The method changes current bitmap, it blends pixels in the rectangle area defined by left-top coordinates of the current bitmap and size of source bitmap using alpha channel from source bitmap (it should be RGBA or BGRA format). Area of replacement must be completely within size of current bitmap.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

pBmp Parameter

Type: IBitmapData.

The source bitmap.

X,Y Parameter

Type: sx_uint.

The coordinates of replacement area.

FillPixels Method

The method changes current bitmap, it sets pixels color in the rectangle area defined by left-top coordinates of the current bitmap and specified color. Area of filling must be completely within size of current bitmap.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

ClearPixels Method

The method changes current bitmap, it sets pixels alpha channel in the rectangle area defined by left-top coordinates of the current bitmap and specified alpha value. Area of filling must be completely within size of current bitmap.

The method produces the exceptions if failed, see ErrorCode property of exception for details.

IPaletteData Interface

The interface represents palette data for indexed formats (see PixelFormat property) of IBitmapData.

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.

Enumerations

PixelFormat Enumeration

This enumeration defines available pixel formats for IBitmapData objects.

PixelFormat_BlackWhite

1 bit per pixel, palette has 2 colors – black and white.

PixelFormat_Grayscale2

2 bit per pixel, palette has 4 grayscale colors.

PixelFormat_Grayscale4

4 bit per pixel, palette has 16 grayscale colors.

PixelFormat_Grayscale

8 bit per pixel, palette has 256 grayscale colors.

PixelFormat_Monochrome

1 bit per pixel, palette has 2 colors.

PixelFormat_Indexed2

2 bit per pixel, palette has 4 colors.

PixelFormat_Indexed4

4 bit per pixel, palette has 16 colors.

PixelFormat_Indexed

8 bit per pixel, palette has 256 colors.

PixelFormat_RGB

24 bit per pixel, 8 bits for every color, order is RGB.

PixelFormat_RGBA

32 bit per pixel, 8 bits for every color and alpha channel, order is RGBA.

PixelFormat_BGR

24 bit per pixel, 8 bits for every color, order is BGR.

PixelFormat_BGRA

32 bit per pixel, 8 bits for every color and alpha channel, order is RGBA.

PixelFormat_CMYK

32 bit per pixel, 8 bits for every color, CMYK color space.

ImageFormat Enumeration

This enumeration defines file formats for saving of IBitmapData objects.

ImageFormat_BMP

ImageFormat_JPEG

ImageFormat_PNG

ImageFormat_TIFF

ImageFormat_GIF

ImageFormat_HDPhoto

ImageInterpolation Enumeration

This enumeration defines interpolation methods used for resizing of IBitmapData objects.

ImageInterpolation_neighbor

ImageInterpolation_linear

ImageInterpolation_bilinear

ImageInterpolation_bicubic

ImageInterpolation_blackman

ImageInterpolation_catmul_rom

ImageInterpolation_lanzcos

ImageInterpolation_gauss

ImageInterpolation_spline

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.

> **C++** 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 completelly 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 destroed 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 cannot 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 cannot 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.

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 cannot 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 completelly 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 horisontal 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 annoatations. 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 immediatelly 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 appearences of all field widgets in the form.

bClear Parameter

Type: sx_bool.

Set this parameter to true, if you want remove existing appearences 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 breaked 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 apperance was adjusted
- pdfa_mess_portfolio - error, cannot 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 (cannot 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 cannot 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 adjuested

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_freetext

annot_type_line

annot_type_square

annot_type_circle

annot_type_polygon

annot_type_polyline

annot_type_highligh

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::EInvoice Namespace

The PDF Xpansion SDK provides complete functionality for processing PDF/A based (ZUGFeRD, Factur-X, Order-X) and XML based (XRechnung, ZUGFeRD, Factur-X, Order-X, Peppol BIS) documents. Processing includes reading data from incoming e-invoices/e-orders, creating outgoing e-invoices/e-orders, visualization of XML-invoices and many other related functions.

Electronic invoice standards and legal basis

The European Parliament and Council voted the Directive 2014/55/EU on electronic invoicing in public procurement on **16 April 2014**. This Directive calls for the definition of a common **European standard on electronic invoicing (EN 16931)** at the semantic level, and additional standardisation deliverables which will enhance interoperability at the syntax level.

Die gesetzlichen Grundlagen zur Einführung der E-Rechnung in Deutschland finden Sie hier.

EN 16931 is the official European standard for electronic invoicing, which defines a common semantic data model for the core content of an electronic invoice. Established by the European Committee for Standardization (CEN), the standard aims to ensure interoperability and simplify cross-border trade within the EU by mandating a structured, machine-readable format for e-invoices, particularly in public procurement.

EN 16931 defines essential data elements and supports various syntaxes like UN/CEFACT Cross Industry Invoice (CII) and ISO/IEC 19845 (UBL 2.1), often implemented via national CIUS (Country-specific Implementation Units) like Germany's **XRechnung** or **ZUGFeRD**, France's **Factur-X** and transmission networks like **Peppol**.

CIUS versions suported by SDK actually:
- XRechnung 3.0.2
- ZUGFeRD 2.4
- FacturX 1.0
- Peppol BIS Billing 3.0

Please take in account that the **XRechnung** and **ZUGFeRD** formats have their own extensions that enabling industry-specific requirements such as the hierarchical organization of invoice items (sub-items). It complements the core model of the **EN 16931** and allows for the mapping of more complex data structures that go beyond the basic standard, in order to meet the requirements of various sectors.

Ultimately, the e-invoice can be presented as a PDF/A based (ZUGFeRD, Factur-X and Order-X) or XML based (XRechnung and Peppol BIS) file.

PDF/A-3 based e-invoice/e-order

The PDF/A based electronic invoice (**ZUGFeRD**, **Factur-X**) or order (**Order-X**) is PDF file that conform to the PDF/A-3 standard, include a standard set of metadata and embedded XML file that conform to XRechnung, ZUGFeRD, Factur-X or Order-X standard. One or more pages in such PDF file represents invoice or order data in a classic form for human reading. The data presented on the pages of the document must correspond to the data contained in the embedded XML, which is intended for machine processing. However, the embedded XML may contain much more or more detailed data than that displayed on the pages.

Important!
Starting with version ZUGFeRD 2.1, the ZUGFeRD standard fully follows the specification of the Factur-X 1.0 standard, including format identification. Thus, it not an independent format anymore and it is technically impossible to identify the files of format ZUGFeRD 2.1 and higher – they are Factur-X 1.0 files only.

Create new outgoing e-invoice or e-order

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 conform to XRechnung (UN/CEFACT CII syntax only), ZUGFeRD, Factur-X or Order-X standard and represents structured invoice/order data.

If you do not have a ready XML-formatted e-invoice/e-order, you can use our SDK to produce such XML file "on the fly".

If you do not have a ready PDF file with your invoice in the traditional form, you can use the **visualization** possibility of SDK (it creates PDF file from an XML e-invoice using the design template.

At first you need to create an object with IInvoiceDocument interface (using IAppFactory interface) which provides all nessesary functionality. Then call CreateFromPDF method 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/order
- conformity of root node
- conformity of invoice/order 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. After the successful creation of PDF/A based invoice/order, you can use the Save method to save it to the file. Please see our "e-invoice" example for more details.

Open incoming e-invoice or e-order, extract XML data

If you get a PDF file, supposedly with an electronic invoice or order, you need to detect and validate the conformity of this file to one of the e-invoice/e-order standards: ZUGFeRD, FACTUR-X or Order-X. At first you need to create an object with IInvoiceDocument interface (using IAppFactory interface) which provides all nessesary functionality. Then call Open method for loading and validation of the PDF/A based e-invoice/e-order.

Validation at this step includes:

- conformity of PDF metadata to PDF/A-3 and e-invoice/e-order standards
- embedding conformity of XML e-invoice/e-order
- optional and simple check of XML e-invoice/e-order (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 e-invoice/e-order using **Standard** and **Profile** properties. Use the GetInvoice for extracting of embedded XML file with electronis invoice or order. Please see our “e-invoice” example for more details.

You can use our SDK for further processing of this XML file. Optionally you can display pages of this PDF invoice/order using our PDF viewer or print it using **GetViewableDocument** or **GetPrintProvider** methods of IBaseDocument interface.

XML based e-invoice/e-order

The XML based electronic invoice or order is XML file that is intended for machine processing of e-invoice/e-order data. Such XML files can be used on their own or embedded into the PDF/A based (ZUGFeRD, Factur-X, Order-X) e-invoice/e-order. For processing the XML e-invoices you need to create an object with IInvoiceData interface (see IInvoiceDocument and IAppFactory interfaces) which provides all nessesary functionality.

PDF Xpansion SDK supports all published versions of following standards

- electronic invoices
  - ZUGFeRD
  - Factur-X
  - XRechnung (UN/CEFACT CII and UBL syntaxes), including CreditNote
  - Peppol BIS Billing (Invoice and CreditNote)
- electronic orders
  - Order-X
  - Peppol BIS Ordering

The **IInvoiceData** interface provides this functionality:

- Create outgoing e-invoices
- Read data from incoming e-invoices
- Edit e-invoice data in your interactive editor
- Adjust standard, version and profile of e-invoice
- Visualize the data of e-invoice (create PDF file from an XML e-invoice using the design template)

You can use IOrderX interface and IOrderBIS interface for processing the XML e-orders.

Please see our “e-invoice” and “e-order” examples for more details.

E-invoice standard extensions

The **XRechnung** and **ZUGFeRD** formats have their own extensions that enabling industry-specific requirements such as the hierarchical organization of invoice items (sub-items). It complements the core model of the **EN 16931** and allows for the mapping of more complex data structures that go beyond the basic standard, in order to meet the requirements of various sectors.

Extension XRechnung

The XRechnung extension should be used when the content of the “EN 16931”-compliant XRechnung is insufficient to submit the required information in the electronic invoice to public contracting authorities. Therefore, the content of the XRechnung extension is optional for invoice issuers. The XRechnung extension complies with the requirements of “EN 16931-5” and is therefore conformant according to “EN 16931”.

XRechnung extension uses its own specification identification in XML. Therefore, within SDK you can identify an extended invoice or create an extended invoice using **einv_profile_extended** profile value. Please note, that **XRechnung extension can be saved in UBL syntax only**, because non-standard elements are not defined in UN/CEFACT CII syntax.

Actually, these elements are allowed within of Extension XRechnung only:

- hierarchical organization of invoice items (sub-items) [BG-DEX-01]
- details about third-party receivables (note, that these receivables only affect the amount payable, they are not part of the actual invoice) [BG-DEX-09]

Extended profile of ZUGFeRD

The ZUGFeRD (and Factur-X) standard incorporates the profile EXTENDED (**einv_profile_extended** within SDK API), which is still based on the UN/CEFACT CII D22B syntax, integrating additional business data and the ability to produce multi-delivery invoices.

Compared to local **XRechnung extension** variations, the extended profile of ZUGFeRD significantly changes both the set of acceptable invoice elements and the cardinality or acceptable values in many cases of standard elements. More detailed information can be found in the original ZUGFeRD documentation of the extended profile. PDF Xpansion SDK completely covers the possibilities of extended profile, in other words, you can get or set any elements allowed in extended profile also.

Hierarchical organization of invoice items (sub-items) in ZUGFeRD

Unlike the **XRechnung extension**, where the sub-items are syntactically implemented based on the hierarchy of the corresponding XML elements, in the **ZUGFeRD** the list of invoice positions in XML always remains linear (GetSubItem method of ILineItem cannot be used), therefore, for processing hierarchical organization of invoice items in the extended profile of the ZUGFeRD, it is necessary to use two additional properties of the ILineDoc interface (**Parent** [BT-X-304] and **StatusReasonCode** [BT-X-8]), as well as its **ID** property.

Use **Parent** property for exploring or setting "vertical" relations between invoice items - the value of this property must be equal to value **ID** property of ascending item in the hierarchy.

Use **StatusReasonCode** for getting or setting of the calculation rules for this item:

INFORMATION value excludes this item and its sub-items from the total's calculation

GROUP value declares sub-total position, this item should have at least one calculable sub-item

DETAIL (or absent value) declares normal (calculable) position, this item can have the information sub-items only

Important!
Invoice items with StatusReasonCode = GROUP or INFORMATION remove some requirements of the EN 16931 standard for interfaces, f.e. net price or billed quantity can be omitted.

E-invoice attachments

Attachments in e-invoices are supporting documents (PDF, PNG, JPEG, XLSX, ODS, CSV, and XML) embedded directly into the XML data structure using Base64 encoding, allowing, for example, for contract details, or time sheets to be submitted with invoices. Synonyms or related terms include supporting documents, embedded files, or binary objects. You can find more detailed information about processing attachments for XML-based e-invoices in the description of GetAdditionalDoc of IAgreement interface.

PDF/A-based e-invoices allow attachments to be embedded directly into the internal structure of PDF document, while the XML portion of the e-invoice only contains a reference to the attachment. This dramatically reduces the common size of the PDF/A-based e-invoice. Please use GetAttachment and SetAttachment methods of IInvoiceDocument for processing such attachments.

IInvoiceDocument Interface

The interface provides functionality for processing PDF/A-3 based e-invoices and e-orders. Using **CreateInvoice** method you can create an object with IInvoiceData interface for processing XML based e-invoices.

This interface is derived from IBaseDocument, so you can load this PDF document to our PDF viewer. You can create a new instance of an object using the IAppFactory interface.

C++ We highly recommend you use an instance of an ObjPtr template class where you can use an EInvoice::IInvoiceDocument pointer in a safe manner.

Standard Property

Type: einv_standard. Access: readonly.

The property retrieves standard of e-invoice or e-order.

Profile Property

Type: einv_profile. Access: readonly.

The property retrieves profile of e-invoice.

Title Property

Type: IStr. Access: full. This is property of PDF document only, it specifies the title of document.

Subject Property

Type: IStr. Access: full. This is property of PDF document only, it specifies the subject of document.

Author Property

Type: IStr. Access: full. This is property of PDF document only, it specifies the author (person or company) of document.

Creator Property

Type: IStr. Access: full. This is property of PDF document only, it specifies the creator of document.

Open Method

The method loads the PDF/A-3 e-invoice or e-order from the stream or file. It returns validation status of loaded document.

The **einv_status_valid** status confirms that the opened document complies with the e-invoice/e-order standard (you can get it using **Standard** property). Use the **GetInvoice** method for extracting of embedded XML stream with electronis invoice or order. You can process the XML invoices using IInvoiceData interface and the XML orders using IOrderX interface or IOrderBIS interface. Optionally you can display pages of this PDF invoice/order using our PDF viewer or print it using **GetViewableDocument** or **GetPrintProvider** methods of IBaseDocument interface.

You can open PDF file with the result of visualization of e-invoice using this method (the status must be **einv_status_autogen** in this case), but exclusively for viewing in our PDF viewer or printing on a printer.

If you get error opening status for some files, but you still want to extract embedded XML stream with electronis invoice or order from these files, then you can call the method again with the option **einv_options_extract_xml**. If method returns **einv_status_nonconform** status, you can use the **GetInvoice** method for extracting of embedded XML stream.

Stream Parameter

Type: ISequentialStream.

The document data stream. Please reset stream before load.

Opt Parameter

Type: sx_flags.

The parameter specifies the options of opening:

- einv_options_check_xml – method must check the embedded XML invoice or order additionally, for compliance with the standard declared in the metadata
- einv_options_check_pdfa – method must check the PDF additionally, for compliance with the PDF/A standard
- einv_options_extract_xml – this option allows to skip the validation of PDF metadata and extract the embedded XML invoice or order, if you want to accept an incoming e-invoice/e-order with the error validation status

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.

CreateFromPDF Method

The method creates the PDF/A-3 e-invoice or e-order from a PDF file that represents invoice or order data in a classic form and XML file that conform to ZUGFeRD, Factur-X, XRechnung (UN/CEFACT CII syntax only) or Order-X standard. **The standard of e-invoice or e-order must be declared using opt parameter** (by default method creates ZUGFeRD 1.0, which is obsolete and no longer used). 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.

Important!
Starting with version ZUGFeRD 2.1, the ZUGFeRD standard fully follows the specification of the Factur-X 1.0 standard, including format identification. Thus, it not an independent format anymore and it is technically impossible to identify the files of format ZUGFeRD 2.1 and higher – they are Factur-X 1.0 files only.

pInvoice Parameter

Type: ISequentialStream.

The XML stream or file. Please reset stream before load.

pPDF Parameter

Type: ISequentialStream.

The PDF stream or file. Please reset stream before load.

Opt Parameter

Type: sx_flags.

The parameter declares standard of e-invoice/e-order (if the declared standard does not correspond to the standard of XML file, then method throws an error):

- einv_options_facturx – Factur-X 1.0 (ZUGFeRD 2.x)
- einv_options_xrechnung30 - XRechnung 3.0
- einv_options_xrechnung23 - XRechnung 2.3
- einv_options_xrechnung22 - XRechnung 2.2
- einv_options_xrechnung21 - XRechnung 2.1
- einv_options_xrechnung20 - XRechnung 2.0
- einv_options_zugferd20 - ZUGFeRD 2.0
- einv_options_orderx - Order-X 1.0

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

Save Method

The method saves the PDF/A-3 e-invoice or e-order to the stream or file. It can be used after successful calling of CreateFromPDF method only.

Stream Parameter

Type: ISequentialStream.

The stream or file for save.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetAttachment Method

The method extracts stream with additional document referenced in an e-invoice element and embedded to this PDF/A document. It can be used after successful calling of Open method only.

Stream Parameter

Type: ISequentialStream.

The stream or file for save.

pFilename Parameter

Type: sx_str.

The filename of an additional document (as it specified in IAttachment element).

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

SetAttachment Method

The method embeds stream with additional document referenced in an e-invoice element to this PDF/A document. It can be used after successful calling of CreateFromPDF method only.

Stream Parameter

Type: ISequentialStream.

The stream or file with additional document.

pFilename Parameter

Type: sx_str.

The filename of an additional document (as it specified in IAttachment element).

pMime Parameter

Type: sx_str.

The MIME type of an additional document.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetAttachmentName Method

The method allows to enumerate all embedded additional documents to this PDF/A document, it returns filename of requested document or null.

ind Parameter

Type: sx_uint.

The index of an embedded additional document. Can be greater than or equal to 0.

GetInvoice Method

The method extracts embedded stream with XML invoice or order. It can be used after successful calling of Open method only.

Stream Parameter

Type: ISequentialStream.

The stream or file for save.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

CreateInvoice Method

The method creates new object for processing of XML based e-invoices, it returns IInvoiceData interface. Please note, it is blank object which can be used for opening an existing XML e-invoice, for creating new XML e-invoice or for other provided operations with XML e-invoices.

> **C++** We highly recommend you use an instance of an ObjPtr template class where you can use an EInvoice::IInvoiceData pointer in a safe manner.

CreateOrderX Method

The method creates new object for processing of XML based Order-X, it returns IOrderX interface.

> **C++** We highly recommend you use an instance of an ObjPtr template class where you can use an EInvoice::IOrderX pointer in a safe manner.

CreateOrderBIS Method

The method creates new object for processing of Peppol BIS Order, it returns IOrderBIS interface.

C++ We highly recommend you use an instance of an ObjPtr template class where you can use an EInvoice::IOrderBIS pointer in a safe manner.

IInvoiceData Interface

The interface provides functionality for processing of XML based electronic invoices (XRechnung, ZUGFeRD and Factur-X):

- Create outgoing e-invoices
- Read data from incoming e-invoices
- Edit e-invoice data in your interactive editor
- Adjust standard, version and profile of e-invoice
- Visualize the data of e-invoice (create PDF file from an XML e-invoice using the design template)

The SDK API for processing of XML e-invoices is a group of objects that form a multi-level e-invoice structure (structure of underlying objects) at the first level of which are the IInvoiceDescription and IInvoiceTransaction objects (interfaces). This structure **fully corresponds** to the structure of the electronic invoice in accordance with the **EN16931 standard, UN/CEFACT CII syntax**. For example, **IInvoiceDescription** represents **ExchangedDocument** element, **IInvoiceTransaction** represents **SupplyChainTradeTransaction** element.

The XML e-invoices in **UBL syntax** processed by SDK without any restrictions - the structure of electronic invoice in UBL syntax is transformed without loss between UN/CEFACT CII and UBL syntaxes while opening or saving an XML file (see OpenData and SaveData methods). This way your application works with all e-invoices only in the UN/CEFACT CII paradigm, regardless of the physical syntax of the e-invoice. This applies to both incoming and outgoing invoices.

The SDK API covers elements outside the EN16931 scheme also – all additional elements from the extended profile in ZUGFeRD standard, as well as elements from Extension XRechnung: SUB INVOICE LINE [BG-DEX-01] and THIRD PARTY PAYMENT [BG-DEX-09].

Important!
This documentation is primarily a description of the SDK API, and in addition provides reference information regarding individual elements of various electronic invoice/order standards, see markup in [BT-00] format.
Thus, it **DOES NOT REPLACE** any of the supported standards, and you should be guided in all matters concerning the requirements and capabilities of a particular standard primarily by the official specification of the standard (and version) used.

It can be imported or exported some other national standards of electronic invoices also.

You can create a new instance of this object using the IInvoiceDocument::CreateInvoice method.

C++ We highly recommend you use an instance of an ObjPtr template class where you can use an EInvoice::IInvoiceData pointer in a safe manner.

Standard Property

Type: einv_standard. Access: readonly.

This property is a setting for the SDK functionality and is not an element of the invoice.

The standard (inc. version) of e-invoice.

Profile Property

Type: einv_profile. Access: readonly.

This property is a setting for the SDK functionality and is not an element of the invoice.

The profile of e-invoice of CII syntax. For all XRechnung e-invoices it can be “einv_profile_EN16931” (EN16931 conform) or “einv_profile_extended” (Extension).

BusinessProcess Property

Type: sx_str. Access: full. [BT-23]

The property defines the business process identifier of invoice. It can be used within all standards and profiles. **It is required property for XRechnung invoices starting from verson 3.0 and higher**.

Description Property

Type: IInvoiceDescription. [BT-1]

The property is an object which provides general properties for the whole e-invoice such as invoice number, type code of invoice, issue date, notes, etc. This object represents the first level of multi-level e-invoice structure.

Transaction Property

Type: IInvoiceTransaction. [BG-25]

The property is an object which represents main parts of e-invoice – structured information about all invoice elements: individual invoice positions, trade agreement, trade delivery and trade settlement details. This object represents the first level of multi-level e-invoice structure.

Test Property

Type: sx_bool. Access: full. [BT-X-1]

The property declares a test e-invoice. It can be used within extended profile of CII syntax only.

AmountDecimalPlaces Property

Type: sx_uint. Access: full.

This property is a setting for the SDK functionality and is not an element of the invoice.

The property defines the max number of decimals of amount values when creating invoices. The default number is 2. Please note that amount values of IMonetarySummation in accordance with e-invoice standards must be rounded to two decimals and if you change AmountDecimalPlaces property, you need to round all setted IMonetarySummation values yourself.

RecognizeStandard Method

The method recognizes the standard of electronic invoice from XML stream or file. It returns einv_standard. Also, the method does not load invoice data, see **Open/OpenData** methods for this.

Please note: This method does not perform e-invoice validation and does not guarantee that adequate and/or complete invoice data will be obtained from this XML. The method checks only the presence of a correct identifier of one of the supported standards, as well as the presence of the required namespaces of the declared standard in the XML. The conformity of the XML structure to the given standard and the presence of the required subelements are checked only by the **Open** and **OpenData** methods.

Stream Parameter

Type: ISequentialStream.

The e-invoice stream or file. Please reset stream before load.

Open Method

The method loads an electronic invoice from XML stream or file. **The invoice must be complete and conform with UN/CEFACT CII**, otherwise method fails. You can use **OpenData** method for opening non-complete/non-conforme e-invoices, e-invoices in UBL syntax or for import of e-invoices (any supported standard).

The Open method returns validation_status of opened e-invoice. More detailed information about errors or non-conformities encountered during the e-invoice loading you can get using callback interface **IInvoiceValidation**. Set your handler using **SetValidationHandler** method before calling this method.

You can get the standard and profile of loaded e-invoice using **Standard** and **Profile** properties. The e-invoice opened with method is readonly, you can not change any elements of it.

Stream Parameter

Type: ISequentialStream.

The e-invoice stream or file. Please reset stream before load.

St Parameter

Type: einv_standard.

The parameter specifies expected standard of e-invoice (UN/CEFACT CII standards only).

Specify **einv_standard_unknown** value for opening of any standard that conform with UN/CEFACT CII.

The method throws an error, if e-invoice corresponds to other standard.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

OpenData Method

The method loads an electronic invoice from XML stream or file. The e-invoice can be in any supported standard or syntax - the structure of electronic invoice in UBL syntax is transformed without loss between UN/CEFACT CII and UBL syntaxes while opening XML data. This method allows to load partially filled e-invoices for completing of filling or extracting of existing data.

Method returns import status of loaded e-invoice. More detailed information about errors or non-conformities encountered during the e-invoice loading you can get using callback interface **IInvoiceValidation**. Set your handler using **SetValidationHandler** method before calling this method.

You can get the standard and profile of loaded e-invoice using **Standard** and **Profile** properties.

Stream Parameter

Type: ISequentialStream.

The e-invoice stream or file. Please reset stream before load.

Opt Parameter

Type: sx_flags.

The parameter specifies the options of opening:

- einv_options_allow_cdata – this option allows support of CDATA elements in XML
- einv_options_xrechnung30, einv_options_xrechnung23, einv_options_xrechnung22, einv_options_xrechnung21, einv_options_xrechnung20, einv_options_xrechnung – using one of these options you can specify preferred version of XRechnung (UBL e-invoices only)
- einv_options_ubl_codes - adjust the used codes between UN/CEFACT CII and UBL syntaxes automatically, for example BT-8 values UNTDID2475 <-> UNTDID2005 (UBL e-invoices only)

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

Create Method

The method creates new empty e-invoice including the first level objects - IInvoiceDescription and IInvoiceTransaction. In turn, **IInvoiceTransaction** has IAgreement, IDelivery and ISettlement subitems also. Fill complete data of new e-invoice, then call **Save** method. You can use **SaveData** method for saving non-complete e-invoices, e-invoices in UBL syntax or for export of e-invoices (any supported standard).

St Parameter

Type: einv_standard.

The parameter specifies the standard of e-invoice to be created (UN/CEFACT CII standards only). If you want to create UBL e-invoice, you still need to create it as an UN/CEFACT CII XRechnung (any version, for example **einv_options_xrechnung30**) than you can save it in UBL syntax using **SaveData** method.

Pr Parameter

Type: einv_profile.

The parameter specifies the necessary profile of e-invoice. Please note, some standards allow not all profiles, use **einv_profile_extended** for Extension XRechnung.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

Save Method

The method saves an electronic invoice in UN/CEFACT CII syntax to the XML stream or file. **The invoice must be complete – all required elements and properties must be filled**, otherwise method fails. You can use **SaveData** method for saving non-complete e-invoices, e-invoices in UBL syntax or for export of e-invoices (any supported standard). You can check completeness of e-invoice by calling CheckRequirements method before saving. The method fails if you try to save an e-invoice opened with Open method.

Stream Parameter

Type: ISequentialStream.

The e-invoice stream or file for saving.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

SaveData Method

The method saves an electronic invoice to the XML stream or file. The e-invoice can be saved in UBL syntax (any XRechnung standard only) - the structure of electronic invoice is transformed without loss between UN/CEFACT CII and UBL syntaxes while saving XML data. This method allows to save partially filled e-invoices and export to supported national standards of electronic invoices also.

You can check completeness of e-invoice by calling CheckRequirements method before saving.

Stream Parameter

Type: ISequentialStream.

The e-invoice stream or file for saving.

St Parameter

Type: einv_standard.

The parameter specifies the standard of e-invoice to be saved. Can be used the next values only:

- einv_standard_cii_family – save in actual standard and syntax
- einv_standard_ubl_xr30 – save in UBL syntax (XRechnung standards only)
- einv_standard_ubl_cn30 – save as XRechnung CreditNote (XRechnung standards only)
- einv_standard_ubl_bis30 – save as Peppol BIS Invoice (XRechnung standards only)
- einv_standard_ubl_biscn30 – save as Peppol BIS CreditNote (XRechnung standards only)

Use **Adjust** method if you want to change the standard within UN/CEFACT CII family.

Opt Parameter

Type: sx_flags.

The parameter specifies the options of opening:

- einv_options_ubl_codes - adjust the used codes between UN/CEFACT CII and UBL syntaxes automatically, for example BT-8 values UNTDID2475 <-> UNTDID2005 (UBL e-invoices only)

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

CheckRequirements Method

The method checks completeness of e-invoice - presence of all required elements. It returns false, if e-invoice is non-complete and you cannot call **Save** method. More detailed information about missed elements you can get using callback interface **IInvoiceValidation**. Set your handler using **SetValidationHandler** method before calling this method.

Please note: This method is not a full-featured standards validator and does not check that properties values correspond to the code lists defined by the standards, and the business rules of the standards are not validated also.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

Adjust Method

The method allows to change standard and profile of an electronic invoice within UN/CEFACT CII family. The new profile can be same or upgraded, profile degradation is restricted.

Please use **SaveData** method for saving e-invoices in UBL syntax or for export of e-invoices (any supported standard).

St Parameter

Type: einv_standard.

The parameter specifies the new standard of e-invoice - UN/CEFACT CII standards only.

Pr Parameter

Type: einv_profile.

The parameter specifies the new profile of e-invoice. Please note, some standards allow not all profiles, use **einv_profile_extended** for Extension XRechnung.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

ExportPDF Method

The method allows to visualize the data of an XML e-invoice in the human readable form. It uses a prepared design template file which defines graphic representation of invoice elements at the pages of PDF file, which is the result of the method's fucntionaliy. You can display pages this PDF file using our PDF viewer, print it using of IInvoiceDocument functionality or use these PDFs for creating PDF/A based e-invoices Factur-X (ZUGFeRD 2.x). And finally, you can use such PDF files within other software, archiving or sending by e-mail.

One design template file is paced in the **Redist** folder of SDK (E-Rechnungsprotokoll.invtpl). This template represents complete information about all elements and data of the e-invoice in the form of a tabular report. You can find some alternative templates, including designs in the form of classic invoices with main information from the e-invoice, on the Internet page of our product Perfect PDF 12. Please contact us if you need the customization of any template.

Please note, e-invoice must be opened (use Open or OpenDate methods) or created before calling this method.

pTemplate Parameter

Type: ISequentialStream.

The design template stream or file.

pPDF Parameter

Type: ISequentialStream.

The stream or file for saving of generated PDF document.

Opt Parameter

Type: sx_flags.

The parameter specifies the options of opening:

- einv_options_pdfa_3b – generate PDF/A document instead of PDF
- einv_options_landscape – create pages in landscape orientation

ExportPDF Method

The method allows to visualize the content of any XML in the human readable form. It uses a prepared design template file which defines graphic representation of XML elements at the pages of PDF file, which is the result of the method's fucntionaliy. A design template file is paced in the **Redist** folder of SDK (E-Rechnungsprotokoll.invtpl).

pXML Parameter

Type: ISequentialStream.

The XML stream or file to be visualized.

pTemplate Parameter

Type: ISequentialStream.

The design template stream or file.

pPDF Parameter

Type: ISequentialStream.

The stream or file for saving of generated PDF document.

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: IInvoiceValidation.

The callback handler. The IInvoiceValidation interface must implemented by client application.

Enumeration conversion methods

Use the following methods for conversion between SDK enumeration types and string values: ToCurrCode3, FromCurrCode3, ToCountryCode2, FromCountryCode2, ToLangCode2, FromLangCode2, ToAllowanceCode, FromAllowanceCode, ToTaxCode, FromTaxCode.

IInvoiceDescription Interface

This interface provides general properties for the whole invoice such as invoice number, type code of invoice, issue date, notes, etc. It is property of IInvoiceData interface.

Important!
This documentation is primarily a description of the SDK API, and in addition provides reference information regarding individual elements of various electronic invoice/order standards.
Thus, it **DOES NOT REPLACE** any of the supported standards, and you should be guided in all matters concerning the requirements and capabilities of a particular standard primarily by the official specification of the standard (and version) used.

This interface and all subinterfaces allow to change the created or loaded using **OpenData** method invoices only! The invoices opened with **Open** method are completely readonly – you cannot set any properties or create subobjects using "c" parameter of methods with value "true".

ID Property

Type: sx_str. Access: full. [BT-1]

A unique identification of the invoice. The sequential number required in Article 226(2) of the directive 2006/112/EC [2], to uniquely identify the invoice within the business context, time-frame, operating systems and records of the seller. It may consist of one or more number sequences which also may contain alpha-numeric characters. An identification scheme must not be used. **It is required property within all standards and profiles.**

Name Property

Type: sx_str. Access: full. [BT-X-2]

Free text to identify the document type, e.g. "Invoice". It can be used within extended profile of CII syntax only.

TypeCode Property

Type: sx_uint. Access: full. [BT-3]

A code specifying the functional type of the invoice according to the UNTDID 1001. Most typical code examples: 380 (commercial invoice), 381 (credit note), 384 (corrected invoice), 386 (advance payment invoice). **It is required property within all standards and profiles.** It is *cbc:InvoiceTypeCode* node in UBL syntax.

IssueTime Property

Type: IDateTime. Access: full. [BT-2]

The date when the Invoice was issued. **It is required property within all standards and profiles.** It is *cbc:IssueDate* node in UBL syntax. The date format is fixed to "YYYY-MM-DD" in UBL syntax.

Copy Property

Type: sx_bool. Access: full. [BT-X-3]

Indicator “Original/Copy”. Should be **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 used within extended profile of CII syntax only.

Language Property

Type: **LangCode2**. Access: full. [BT-X-4]

Language designation according to the ISO 639-1 (Alpha-2 code space). Use **ToLangCode2** and **FromLangCode2** methods of IInvoiceData for conversion between **LangCode2** type and string value. It can be used within extended profile of CII syntax only.

GetNote Method

Return value: INote. [BG-1]

A textual note that gives unstructured information that is relevant to the Invoice as a whole. Such as the reason for any correction or assignment note in case the invoice has been factored.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetEffectivePeriod Method

Return value: ITimePeriod. [BT-X-6]

Contractual due date of the invoice. It is used very rarely, if the contractual due date differs from the due date of the payment (e.g. for SEPA direct debits) - within extended profile of CII syntax only. Start, end date/time and description of period cannot be used in this context.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IInvoiceTransaction Interface

The interface represents main parts of invoice – structured information about all invoice parts: individual invoice positions, information about goods and services, prices, reference products, etc. It is property of IInvoiceData interface.

Important!
This documentation is primarily a description of the SDK API, and in addition provides reference information regarding individual elements of various electronic invoice/order standards.
Thus, it **DOES NOT REPLACE** any of the supported standards, and you should be guided in all matters concerning the requirements and capabilities of a particular standard primarily by the official specification of the standard (and version) used.

This interface and all subinterfaces allow to change the created or loaded using **OpenData** method invoices only! The invoices opened with **Open** method are completely readonly – you cannot set any properties or create subobjects using "c" parameter of methods with value "true".

Agreement Property

Type: IAgreement. [BT-10]

Grouping of contract information: the prices of goods and services, information about the order, the corresponding agreement and additional documents. **It is required property within all standards and profiles.**

Delivery Property

Type: IDelivery. [BG-X-22]

Grouping of delivery details: information on the different goods recipient and final goods recipient, information on the delivery event. **It is required property within all standards and profiles.**

Settlement Property

Type: ISettlement. [BG-19]

Grouping of payment and billing information: details on taxation, extra charges and discounts, billing period, accounting references of the buyer. **It is required property within all standards and profiles.**

GetLineItem Method

Return value: ILineItem. [BG-25]

Grouping of invoice position information: item number and codes, free text to the item, product information and classification, country of origin, information about the included products. **At least one**

line item (position) must be specified for any valid invoice within all standards and profiles.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). 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

It is subitem of IInvoiceTransaction interface. This interface inherits from the ILineAgreement interface, which is subitem of ILineItem interface.

BuyerReference Property

Type: sx_str. Access: full. [BT-10]

An identifier assigned by the buyer used for internal routing purposes. **An invoice must have buyer reference or purchase order reference (BT-13).** The property is restricted at invoice position level.

Anmerkung: Im Rahmen des Steuerungsprojekts eRechnung ist mit der so genannten **Leitweg-ID** eine Zuordnungsmöglichkeit entwickelt worden, deren verbindliche Nutzung von Bund und mehreren Ländern vorgegeben wird. Die Leitweg-ID ist prinzipiell für Bund, Länder und Kommunen einsetzbar (B2G, G2G). Für die Darstellung der Leitweg-ID wird das in XRechnung verpflichtende Feld **BuyerReference** benutzt.

GetSeller Method

Return value: IParty. [BG-4]

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. **It is required subitem within all standards and profiles.** The subitem is restricted at invoice position level.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetBuyer Method

Return value: IParty. [BG-7]

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. **It is required subitem within all standards and profiles.** The subitem is restricted at invoice position level.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetSellerTaxRepresentative Method

Return value: IParty. [BG-11]

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. The subitem is restricted at invoice position level.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetEndUser Method

Return value: IParty. [BG-X-18]

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. The subitem is restricted at invoice position level. It can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetDeliveryTerms Method

Return value: IDeliveryTerms. [BG-X-22]

Detailed information on delivery conditions. The subitem is restricted at invoice position level. It can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetSellerOrder Method

Return value: IRefDoc. [BT-14] [BG-X-81]

Detailed information on the order referenced document. The subitem can be used on the invoice level within all standards and profiles and at invoice position level within extended profile of CII syntax.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetBuyerOrder Method

Return value: IRefDoc. [BT-13] [BT-132]

Detailed information on the buyer order referenced document. The subitem can be used within all standards and profiles.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetContract Method

Return value: IRefDoc. [BT-12] [BG-X-2]

Detailed information on the according agreement: agreement number, agreement. The subitem can be used on the invoice level within all standards and profiles and at invoice position level within extended profile of CII syntax.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetAdditionalDoc Method

Return value: IRefDoc. [BG-24] [BT-17] [BT-18] [BT-122] [BG-X-3]

A group of business terms providing information about additional supporting documents substantiating the claims made in the invoice. The additional supporting documents can be used for both referencing a document number which is expected to be known by the receiver, an external document (referenced by a URL) or as an embedded document (such as a timesheet as a PDF file). The additional documents can be used on the invoice level within all standards and profiles and at invoice position level within extended profile of CII syntax.

Please note that BT-17 element (type code is 50) in UBL syntax is saved as

“OriginatorDocumentReference”, unlike other documents saved as “AdditionalDocumentReference”.

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.

An attached document can be embedded as binary object (Base64) directly to the XML invoice (see GetAttachment method of IRefDoc interface). In case of ZUGFeRD/Factur-X invoice it can be alternatively embedded to the PDF invoice and referenced in the XML invoice. This reference (URIID property of IRefDoc interface) consists a relative fragment identifier in the form “#ef=filename” instead of URL, which points to an embedded document “filename” of PDF invoice. Please use GetAttachment and SetAttachment methods of IInvoiceDocument for processing the documents embedded in PDF invoice.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetProcuringProject Method

Return value: IProcProject. [BT-11]

Detailed information on the associated project: project reference (number, etc.), project name. The subitem is restricted at invoice position level.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetGrossProductPrice Method

Return value: IPrice. [BT-148]

The price applied for the goods and services invoiced on the invoice position before subtracting price discount (except for VAT), price-related discount and the number of item units to which the price applies. If the price discount is not defined, then gross price should be equal to the net price (BT-146) or this subitem is not specified at all. This price cannot be negative.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetNetProductPrice Method

Return value: IPrice. [BT-146]

The price applied for the goods and services invoiced on the invoice position (including all surchages and discounts, except for VAT) and the number of item units to which the price applies. This price cannot be negative. It could be included tax for B2C (if type of tax is other than VAT, such as insurance tax or mineral oil tax) specified, but within extended profile of CII syntax only. **It is required subitem within all standards and profiles.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetCustomerOrder Method

Return value: IRefDoc. [BG-X-23] [BG-X-5]

Detailed information on the order referenced document - the order number and the order date. The subitem can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetQuotation Method

Return value: IRefDoc. [BG-X-61] [BG-X-47]

Details of a quotation document reference. The subitem can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetBuyerTaxRepresentative Method

Return value: IParty. [BG-X-54]

Detailed information about the buyer tax representative. The subitem can be used on the invoice level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetSalesAgent Method

Return value: IParty. [BG-X-49]

Detailed information about the sales agent. The subitem is restricted at invoice position level. It can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetBuyerAgent Method

Return value: IParty. [BG-X-62]

Detailed information about the buyer agent. The subitem is restricted at invoice position level. It can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

ILineAgreement Interface

Detailed information on the agreement at invoice position level. It is subitem of ILineItem interface. This is base interface for the IAgreement interface, please see description of both interfaces here.

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.)

It is subitem of IInvoiceTransaction interface. This interface inherits from the ILineDelivery interface, which is subitem of ILineItem interface.

GetBilledQuantity Method

Return value: IQuantity. [BT-129]

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 unit of measure for this subitem, must be the same as the unit code of the position price base quantity (BT-149), if present.**

The subitem is allowed at invoice position level only. **It is required within all standards and profiles.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetChargeFreeQuantity Method

Return value: IQuantity. [BT-X-46]

Information about free of charge goods or services quantity. The unit code defines the measure unit for the quantity. The subitem is allowed at invoice position level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetPackageQuantity Method

Return value: IQuantity. [BT-X-47]

Information about the amount of individual goods packages invoiced in the relevant row. The unit code defines the measure unit for the quantity. The subitem is allowed at invoice position level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetChainConsignment Method

Return value: IChainConsignment. [BG-X-24]

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 subitem can be used on the invoice level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetShipToParty Method

Return value: IParty. [BG-13] [BG-X-7]

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 subitem can be used on the invoice level within all standards and profiles and at invoice position level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetUltimateShipToParty Method

Return value: IParty. [BG-X-27] [BG-X-10]

Detailed information on the deviating final. The subitem can be used on the invoice level and at invoice position level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetShipFromParty Method

Return value: IParty. [BG-X-30]

Identification of the deviating sender. The subitem can be used on the invoice level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetDeliveryChainEvent Method

Return value: IChainEvent. [BT-72] [BT-X-85]

Information about the actual delivery time: date of delivery and achievement in terms of taxability. The subitem can be used on the invoice level and at invoice position level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetDespatchAdvice Method

Return value: IRefDoc. [BT-16] [BG-X-13]

Detailed information on the corresponding despatch advice: reference and advice date. The subitem can be used on the invoice level within all standards and profiles and at invoice position level within extended profile of CII syntax.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetReceivingAdvice Method

Return value: IRefDoc. [BT-15] [BG-X-82]

Detailed information on the associated goods receipt: reference and receipt date. The subitem can be used on the invoice level within all standards and profiles and at invoice position level within extended profile of CII syntax.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetDeliveryNote Method

Return value: IRefDoc. [BT-X-202] [BG-X-83]

Detailed information on the corresponding delivery: reference and delivery note date. The subitem can be used on the invoice level and at invoice position level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

ILineDelivery Interface

Detailed information on delivery at invoice position level. It is subitem of ILineItem interface. This is base interface for the IDelivery interface, please see description of both interfaces here.

ISettlement Interface

Detailed information on the payment and billing:

- Position or invoice totals
- Position or invoice allowances and charges

- Position or invoice billing period
- Invoice VAT breakdown
- Position VAT information
- VAT accounting currency code
- Invoice currency code
- Bank assigned creditor identifier
- Remittance information
- Seller reference number
- Payment terms information
- Payment instructions

It is subitem of IInvoiceTransaction interface. This interface inherits from the ILineSettlement interface, which is subitem of ILineItem interface.

CreditorReference Property

Type: sx_str. Access: full. [BT-90]

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. The property is restricted at invoice position level.

This is element of a group of business terms to specify a direct debit [**BG-19**], other elements are “mandate reference identifie” [**BT-89**] and “debited account identifier” [**BT-91**].

PaymentReference Property

Type: sx_str. Access: full. [BT-83]

A textual value used to establish a link between the payment and the invoice, issued by the seller. Used for creditor's critical reconciliation information. It helps the seller to assign an incoming payment to the relevant payment process. The property is restricted at invoice position level.

Currency Property

Type: **CurrCode3**. Access: full. [BT-5]

Invoice currency code according to the ISO 4217 (Alpha-3 code space). Only one currency shall be used in the invoice, except for the total VAT amount in accounting currency (BT-111) in accordance with article 230 of Directive 2006/112/EC on VAT. **It is required property within all standards and profiles.** The property is restricted at invoice position level. Use **ToCurrCode3** and **FromCurrCode3** methods of IInvoiceData for conversion between **CurrCode3** type and string value.

TaxCurrency Property

Type: **CurrCode3**. Access: full. [BT-6]

VAT accounting currency code according to the ISO 4217 (Alpha-3 code space). Shall be used in combination with the total VAT amount in accounting currency (BT-111) when the VAT accounting currency code differs from the invoice currency code. The property is restricted at invoice position level. Use **ToCurrCode3** and **FromCurrCode3** methods of IInvoiceData for conversion between **CurrCode3** type and string value.

IssuerReference Property

Type: sx_str. Access: full. [BT-X-204]

Given seller reference number for routing purposes after biliteral agreement. The property is restricted at invoice position level. It can be used within extended profile of CII syntax only.

GetInvoicerParty Method

Return value: IParty. [BG-X-33]

Deviating invoicing party. The subitem is restricted at invoice position level. It can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetInvoiceeParty Method

Return value: IParty. [BG-X-36]

Detailed information about the deviating invoice recipient. The subitem is restricted at invoice position level. It can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetPayeeParty Method

Return value: IParty. [BG-10] [BG-X-77]

The role of payee may be fulfilled by another party then the seller, e.g. a factoring service. The subitem can be used on the invoice level within all standards and profiles and at invoice position level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetPayerParty Method

Return value: IParty. [BG-X-73]

Detailed information about the deviating invoice payer. The subitem is restricted at invoice position level. It can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetCurrencyExchange Method

Return value: ICurrencyExchange. [BG-X-41]

Specification of the invoice currency, local currency and exchange rate. The subitem can be used on the invoice level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetPaymentMeans Method

Return value: IPaymentMeans. [BG-16]

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 subitem(s) can be used on the invoice level only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetTax Method

Return value: ITax. [BG-23] [BG-30]

This subitem contains different information at the invoice level and at the invoice position level.

Invoice level: a group of business terms providing information about VAT breakdown by different categories, rates and exemption reasons. **At least one VAT breakdown must be specified for any valid invoice within all standards and profiles.**

Invoice position level: a group of business terms providing information about the VAT applicable for the goods and services invoiced on the invoice position. **At least one VAT subitem must be specified**

for any valid invoice position within all standards and profiles.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetBillingPeriod Method

Return value: ITimePeriod. [BG-14] [BG-26]

Subitem used to indicate when the period covered by the invoice starts and when it ends (at the invoice level) or the invoice period relevant for the current invoice position. Also called delivery period. The subitem can be used within all standards and profiles.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetAllowanceCharge Method

Return value: IAllowanceCharge. [BG-20] [BG-21] [BG-27] [BG-28]

A group of business terms providing information about allowances, charges and taxes other than VAT applicable to the invoice as a whole or to the individual invoice position. The subitem can be used within all standards and profiles.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetLogisticsCharge Method

Return value: ILogisticsCharge. [BG-X-42]

Detailed information on logistics service fees - transport and packaging costs. The subitem can be used on the invoice level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetPaymentTerms Method

Return value: IPaymentTerms. [BT-20]

A textual description of the payment terms that apply to the amount due for payment (including description of possible penalties). The subitem can be used on the invoice level only. It could be multiple subitems within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetMonetarySummation Method

Return value: IMonetarySummation. [BG-22] [BT-131]

Detailed information about the totals - to the invoice as a whole or to the individual invoice position. **It is required subitem within all standards and profiles at the invoice level and at the invoice position level.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetReferencedDoc Method

Return value: IRefDoc. [BG-3] [BT-128] [BG-X-48]

This subitem contains different information at the invoice level and at the invoice position level.

Invoice level: a group of business terms providing information on one or more preceding invoices.

To be used in case:
- a preceding invoice is corrected
- preceding partial invoices are refered to from a final invoice
- preceding pre-payment invoices are refered to from a final invoice

Invoice position level: An identifier for an object on which the invoice position is based, given by the seller. It could be multiple subitems within extended profile of CII syntax only. The extended profile of CII syntax allows one preceding invoice at invoice position level to be specified – please use value -1 of “i” parameter to operate with it.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetReceivableAccount Method

Return value: IAccount. [BT-19] [BT-133]

A refrence that specifies where to book the relevant data into the buyer's financial accounts. The subitem can be used within all standards and profiles. It could be multiple subitems on the invoice level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetAdvancePayment Method

Return value: IAdvancePayment. [BG-X-45]

Included tax for advanced payment. The subitem can be used on the invoice level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetPrepaidPayment Method

Return value: IPrepaidPayment. [BG-DEX-09]

A group of information elements that contain information about third-party claims. Note: third-party claims only affect the amount to be paid, but are not part of the actual invoice. The subitem can be used within Extension XRechnung of UBL syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

ILineSettlement Interface

Detailed information on the payment and billing at invoice position level. It is subitem of ILineItem interface. This is base interface for the ISettlement interface, please see description of both interfaces here.

ILineItem Interface

Grouping of invoice position information (item number and codes, free text to the item, price and quantity, discount and charges, product information and classification, country of origin). It could provide information about subordinate invoice items (sub-items) also, see GetSubItem method of this interface. It is subitem of IInvoiceTransaction interface.

GetDocument Method

Return value: ILineDoc. [BT-126]

Grouping of general position information: identifier of the invoice line item, type of the inovice line item (code), subtype of the invoice line item, detailed information about the free text of the line item. **It is required subitem within all standards and profiles.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetProduct Method

Return value: ILineProduct. [BG-31]

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. **It is required subitem within all standards and profiles.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetAgreement Method

Return value: ILineAgreement. [BG-29]

Detailed information on the price, order details (order number, reference to the order line item, date, referenced agreement (number, position, date), additional referenced document (number, type code, email, binary attachement, date). **It is required subitem within all standards and profiles.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetDelivery Method

Return value: ILineDelivery. [BT-129]

Grouping of delivery details on invoice position level: billed quantity, packaged quantity, free quantity. Unit codes for measurements, deviating recipient and final recipient, delivery documents (delivery bill, advice, delivery notification, etc.). **It is required subitem within all standards and profiles.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetSettlement Method

Return value: ILineSettlement. [BG-30]

Grouping of billing information at invoice position level: applicable trade tax, invoice line billing period, allowances and charges on invoice position level, information on the item totals, additional referenced document, detailed information on the accounting reference. **It is required subitem within all standards and profiles.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetSubItem Method

Return value: ILineItem. [BG-DEX-01]

A group of information elements that contain information about individual subordinate invoice items (sub-items). Note: it is assumed that all subpositions in a hierarchy level (including the associated surcharges and discounts) are subject to the same (sales) tax rate as the parent invoice position. **The subitem can be used within Extension XRechnung of UBL syntax only**.

Note: it is assumed that all sub-positions in a hierarchy level (including the associated surcharges and discounts) are subject to the same tax rate as the parent invoice item (position).

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

ILineDoc Interface

Grouping of general position information - identifier of the invoice position, type and subtype codes of

the position, detailed information about the free text of the position. It is subitem of ILineItem interface.

ID Property

Type: sx_uint. Access: full. [BT-126]

Use **IDStr** property for non-numeric values, type: sx_str.

A unique identification for the individual invoice position within the invoice. It may be number or may contain alpha-numeric characters. An identification scheme must not be used. **It is required property within all standards and profiles.**

Parent Property

Type: sx_str. Access: full. [BT-X-304]

Parent line ID - the value given here refers to the superior line. In this way, a hierarchy tree of invoice positions can be mapped. The property can be used within extended profile of CII syntax only.

StatusCode Property

Type: sx_uint. Access: full. [BT-X-7]

A code specifying type of the inovice position according to the UNTDID 1229. The default code is 39. Indicating whether a position includes the prices which must be considered when calculating the invoice amount, or whether it only contains information. The property can be used within extended profile of CII syntax only.

StatusReasonCode Property

Type: sx_uint. Access: full. [BT-X-8]

A code specifying subtype of the inovice position. Possible values:

- DETAIL - regular invoice position (standard case)
- GROUP - subtotal
- INFORMATION - for information only

The property can be used within extended profile of CII syntax only.

GetNote Method

Return value: INote. [BT-127]

A textual note that gives unstructured information that is relevant to the invoice position. It could be a single item within all standards and profiles or multiple subitems within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

ILineProduct Interface

An aggregation of business terms, which contains information about the invoiced products and services - global ID, an identification of the position assigned by the seller and the buyer, an article’s name and description, product attribute values, detailed information on product classification. It is subitem of ILineItem interface.

Name Property

Type: sx_str. Access: full. [BT-153] [BT-X-18]

A name for an invoice position. **It is required property within all standards and profiles.**

Description Property

Type: sx_str. Access: full. [BT-154] [BT-X-19]

The description allows for describing the invoice position and its features in more detail than the position name. The subitem can be used within all standards and profiles.

GetGlobalID Method

Return value: IGlobalID. [BT-157] [BT-X-15]

An invoice position identifier based on a registered scheme, which shall be identified from the entries of the list published by the ISO/IEC 6523 maintenance agency. The subitem can be used within all standards and profiles.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

ID Property

Type: sx_str. Access: full. [BT-X-305] [BT-X-308]

The product identifier. The property can be used within extended profile of CII syntax only.

SellerAssignedID Property

Type: sx_str. Access: full. [BT-155] [BT-X-16]

An identifier, assigned by the seller, for the invoice position. The property can be used within all standards and profiles.

BuyerAssignedID Property

Type: sx_str. Access: full. [BT-156] [BT-X-17]

An identifier, assigned by the buyer, for the invoice position. The property can be used within all standards and profiles.

IndustryAssignedID Property

Type: sx_str. Access: full. [BT-X-532] [BT-X-309]

Industry assigned product identifier. The property can be used within extended profile of CII syntax only.

ModelID Property

Type: sx_str. Access: full. [BT-X-533]

Model identification of the product. The property can be used within extended profile of CII syntax only. The property is restricted for included referenced products.

BatchID Property

Type: sx_str. Access: full. [BT-X-534]

Batch (lot) identification of the product. The property can be used within extended profile of CII syntax only. The property is restricted for included referenced products.

BrandName Property

Type: sx_str. Access: full. [BT-X-535]

Product brand name. The property can be used within extended profile of CII syntax only. The property is restricted for included referenced products.

ModelName Property

Type: sx_str. Access: full. [BT-X-536]

Product model name. The property can be used within extended profile of CII syntax only. The property is restricted for included referenced products.

Country Property

Type: **CountryCode2**. Access: full. [BT-159]

The code identifying the country from which the product originates. Country code according to the ISO 3166-1 (Alpha-2 code space). Use **ToCountryCode2** and **FromCountryCode2** methods of IInvoiceData for conversion between **CountryCode2** type and string value. The subitem can be used within all standards and profiles.

GetCharacteristic Method

Return value: IProductChar. [BG-32]

A group of business terms providing information about properties of the goods and services invoiced. The subitem can be used within all standards and profiles.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetClassification Method

Return value: IProductClass. [BT-158]

Detailed information on the product classification. The subitem can be used within all standards and profiles.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetInstance Method

Return value: IProductInstance. [BG-X-84]

A product instances. The subitem can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetRefProduct Method

Return value: IProduct. [BG-X-1]

An included product referenced from this trade product. The subitem can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetQuantity Method

Return value: IQuantity. [BT-X-20]

Quantity of included referenced product. The subitem can be used for included referenced product within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IProduct Interface

Detailed information about the product. It is subitem of ILineProduct interface. This is base interface for the ILineProduct interface also, please see description of both interfaces here.

IPrice Interface

The price applied for the goods and services invoiced on the invoice position (except for VAT) and the number of item units to which the price applies. Using this interface, you can define gross price (BT-148) and net price (BT-146). For nuances and differences in determining these prices, see the description of the methods below. Both prices are subitems of the ILineAgreement interface.

GetAmount Method

Return value: IAmount. [BT-148] [BT-146]

The price applied for the goods and services invoiced on the invoice position (except for VAT). In the case of gross price, this amount does not include the price discount specified as an allowance subitem of the price. In the case of net price, this amount includes the price discount if it presents in gross price definition). The amount cannot be negative. **It is required subitem for gross (if present) and net prices within all standards and profiles.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetQuantity Method

Return value: IQuantity. [BT-149]

Item price base quantity - the number of item units to which the price applies. **The unit of measure for this subitem, must be the same as the unit code of the invoiced quantity of position (BT-129).**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetAllowanceCharge Method

Return value: IAllowanceCharge. [BT-147] [BT-X-299]

The total discount subtracted from the gross price to calculate the item net price. Only applies if the discount is provided per unit and if it is not included in the gross price yet. The subitem is available in the case of gross price, it is restricted for net price. It could be specified multiple discounts and surcharges for gross price, but within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetTax Method

Return value: ITax. [BG-X-4]

Included tax for B2C (if type of tax is other than VAT, such as insurance tax or mineral oil tax) specified. This subitem can be used in the net price item within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IAllowanceCharge Interface

A group of business terms providing information about allowances, charges and taxes other than VAT applicable to the invoice as a whole or to the individual invoice position. Also, this interface defines price-related discount within the gross price of invoice position. It is subitem of ISettlement, ILineSettlement and IPrice interfaces.

Charge Property

Type: sx_bool. Access: full.

Use “true” when item describes a charge (surcharge) and “false” when it dscribes allowance (discount). **It is required property within all standards, profiles and use cases of IAllowanceCharge.**

SeqNum Property

Type: sx_uint. Access: full. [BT-X-265] [BT-X-268]

Calculation sequence. The property can be used for allowances/charges on the invoice level within extended profile of CII syntax only.

Percent Property

Type: sx_double. Access: full. [BT-94] [BT-101] [BT-138] [BT-143] [BT-X-34] [BT-X-300]

The percentage that may be used, in conjunction with the allowance/charge base amount, to calculate the allowance/charge amount. The property can be used within all standards, profiles and use cases of IAllowanceCharge exept price discount (in this case within extended profile of CII syntax only).

ReasonCode Property

Type: **allowance_code**. Access: full. [BT-98] [BT-105] [BT-140] [BT-145] [BT-X-313] [BT-X-314]

The reason of the allowance or charge, expressed as a code. For allowances a subset of codelist UNCL5189 is to be used, and for charges codelist UNCL7161 applies. The reason code and the reason as a text shall indicate the same allowance/cahrge reason. Use **ToAllowanceCode** and **FromAllowanceCode** methods of IInvoiceData for conversion between **allowance_code** type and string value. The property can be used within all standards, profiles and use cases of IAllowanceCharge exept price discount (in this case within extended profile of CII syntax only).

Reason Property

Type: sx_uint. Access: full. [BT-97] [BT-104] [BT-139] [BT-144] [BT-X-36] [BT-X-303]

The reason of the allowance or charge, expressed as a text. The property can be used within all standards, profiles and use cases of IAllowanceCharge exept price discount (in this case within extended profile of CII syntax only).

GetBasisAmount Method

Return value: IAmount. [BT-93] [BT-100] [BT-137] [BT-142] [BT-X-35] [BT-X-301]

The base amount that may be used, in conjunction with the allowance/charge percentage, to calculate the allowance/charge amount. The subitem can be used within all standards, profiles and use cases of IAllowanceCharge exept price discount (in this case within extended profile of CII syntax only).

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetActualAmount Method

Return value: IAmount. [BT-92] [BT-99] [BT-136] [BT-141] [BT-147] [BT-302]

The amount of an allowance or a charge, without VAT. Must be rounded to maximum two decimals if it is not price discount. **It is required subitem within all standards, profiles and use cases of IAllowanceCharge.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetCategoryTax Method

Return value: ITax. [BT-95] [BT-102]

A group of business terms providing information about the VAT applicable for an allowance or a charge. **It is required subitem within all standards and profiles in the case of IAllowanceCharge on the invoice level.** The subitem is restricted at invoice position level and for price discount.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetBasisQuantity Method

Return value: IQuantity. [BT-X-266] [BT-X-269]

The basis quantity of an allowance or a charge. The subitem is allowed in the case of IAllowanceCharge on the invoice level within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

ILogisticsCharge Interface

Detailed information on logistics service fees - transport and packaging costs. It is subitem of ISettlement interface, it can be used within extended profile of CII syntax only.

Description Property

Type: sx_str. Access: full. [BT-X-271]

Service fee description. **It is required property.**

GetAmount Method

Return value: IAmount. [BT-X-272]

The amount of a logistics service, without VAT. Must be rounded to maximum two decimals. **It is required subitem.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetTax Method

Return value: ITax. [BT-X-273]

A group of business terms providing information about the VAT applicable for a logistics service. **It is required subitem.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

ITax Interface

This interface describes the trade tax in the different cases - VAT applicable for the goods and services, allowances and charges, VAT breakdowns by different categories, included tax for B2C.

It is subitem of ISettlement, ILineSettlement, IPrice, IAllowanceCharge, ILogisticsCharge and IAdvancePayment interfaces.

TypeCode Property

Type: **tax_code**. Access: full. [BT-X-38] [BT-X-294]

The reason of the allowance or charge, expressed as a code. The property value must be **tax_code_VAT** in all cases except case of included tax for B2C (BG-X-4), such as insurance tax or mineral oil tax, then value is a code specifying the type of the tax according to the UNTDID 5153. **It is required property within all standards, profiles and use cases of ITax.**

CategoryCode Property

Type: **tax_cat**. Access: full. [BT-118] [BT-151] [BT-95] [BT-102] [BT-X-40] [BT-X-273] [BT-X-296]

A coded identification of VAT category of the tax according to the UNTDID 5305. **It is required property within all standards, profiles and use cases of ITax.**

ExemptionReasonCode Property

Type: sx_str. Access: full. [BT-121] [BT-X-41] [BT-X-97] [BT-X-297]

A coded statement of the reason why the amount is exempted from VAT, it must belong to the CEF VATEX code list.

ExemptionReason Property

Type: sx_str. Access: full. [BT-120] [BT-X-39] [BT-X-96] [BT-X-295]

A textual statement of the reason why the amount is exempted from VAT or why no VAT is being charged.

ApplicablePercent Property

Type: sx_double. Access: full. [BT-119] [BT-152] [BT-96] [BT-103] [BT-X-274] [BT-X-298] [BT-X-42]

The VAT rate, represented as percentage that applies to the amount. It is required property for the case of included tax for B2C.

DateCode Property

Type: **date_code**. Access: full. [BT-8]

The code of the date when the VAT becomes accountable for the seller and for the buyer. The code shall distinguish between the following entries of UNTDID 2005:
- invoice issue date
- delivery date, actual
- payment date

The property can be used within all standards and profiles in the case of VAT breakdowns only - it is used if the tax point date is not known when the invoice is issued. The use of BT-8 and BT-7 is mutually exclusive.

GetTaxPoint Method

Return value: IDateTime. [BT-7]

Tax due date, the date when the VAT becomes accountable for the seller and for the buyer in so far as that date can be determined and differs from the date of issue of the invoice, according to the VAT directive. The property can be used within all standards and profiles in the case of VAT breakdowns only. The use of BT-7 and BT-8 is mutually exclusive. This does not apply in Germany - use date of delivery instead.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method

returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetCalculatedAmount Method

Return value: IAmount. [BT-117] [BT-X-37] [BT-X-95] [BT-X-293]

VAT tax amount. In the case of VAT breakdown, it calculated by multiplying the VAT category taxable amount with the VAT category rate for the relevant VAT category. **It is required subitem within all standards and profiles in the cases of VAT breakdowns and included tax for B2C.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetBasisAmount Method

Return value: IAmount. [BT-116]

VAT category taxable amount. It used in the case of VAT breakdown only. The sum of invoice position net amounts minus allowances plus charges on document level which are subject to a specific VAT category code and VAT category rate (if the VAT category rate is applicable). **It is required subitem within all standards and profiles in the case of VAT breakdowns.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetLineTotalBasisAmount Method

Return value: IAmount. [BT-X-262]

Invoice position total basis amount. It can be used in the case of VAT breakdown within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetAllowanceChargeBasisAmount Method

Return value: IAmount. [BT-X-263]

Total amount of charges/allowances on the invoice level. It can be used in the case of VAT breakdown within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method

returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IMonetarySummation Interface

Detailed information about the totals - for the invoice as a whole or for the individual invoice position (usually it is position net amount only, which is always required).

Totals at the invoice level:

- Sum of invoice positions net amount (required)
- Sum of charges on invoice level (optional)
- Sum of allowances on invoice level (optional)
- Invoice total amount without VAT (required)
- Invoice total VAT amount (required)
- Invoice total amount with VAT (required)
- Rounding and prepaid amounts (optional)
- Amount due for payment (required)

Note: while creating an invoice, you can specify your own calculated total amounts (**the standard requires specifying not more than two decimals for these amounts**) or you can use **Calculate** method of this interface after adding all invoice positions with prices and VAT information, also optionally charges and allowances at invoice and/or position level – this method fills all required amounts with actual values.

Important! Please note, when you use trial license, all amount values are randomly multiplied by a value from 2 to 9. This is done for opened and created invoices.

It is subitem of ISettlement and ILineSettlement interfaces.

GetLineTotal Method

Return value: IAmount. [BT-106] [BT-131]

For the individual invoice position, it is net amount (without VAT), i.e. inclusive of allowances and charges of this position as well as other relevant taxes. At the invoice level it is a sum of all invoice positions (∑ BT-131). This subitem can be recalculated using Calculate method. **It is required subitem within all standards and profiles at the invoice level and at the invoice position level.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetChargeTotal Method

Return value: IAmount. [BT-108] [BT-X-327]

Sum of charges on invoice level (BT-99). This subitem can be recalculated using Calculate method. **It is required subitem within all standards and profiles at the invoice level if at least one charge subitem (BG-21) is specified.**

This subitem can be optionally used at the invoice position level, but within extended profile of CII syntax only. In this case it is sum of charges at the invoice position level (BT-141) if at least one charge subitem (BG-28) is specified.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetAllowanceTotal Method

Return value: IAmount. [BT-107] [BT-X-328]

Sum of allowances on invoice level (BT-92). This subitem can be using Calculate method recalculated. **It is required subitem within all standards and profiles at the invoice level if at least one charge subitem (BG-20) is specified.**

This subitem can be optionally used at the invoice position level, but within extended profile of CII syntax only. In this case it is sum of allowances at the invoice position level (BT-136) if at least one charge subitem (BG-27) is specified.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetTaxBasisTotal Method

Return value: IAmount. [BT-109]

The total amount of the invoice without VAT (BT-109 = BT-106 + BT-108 - BT-107). This subitem can be using Calculate method recalculated. **It is required subitem within all standards and profiles at the invoice level.** The subitem is restricted at invoice position level.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetTaxTotal Method

Return value: IAmount. [BT-110] [BT-111] [BT-X-327]

The total VAT amount for the invoice - is the sum of all VAT category (BG-23) tax amounts (∑ BT-117). This subitem can be using Calculate method recalculated. **It is required subitem within all standards**

and profiles at the invoice level. The currency code is required for this subitem.

In the case when the VAT accounting currency (BT-6) is specified and differs from the invoice currency (BT-5), the second subitem of total VAT amount (BT-111) is required. This subitem expressed in the accounting currency accepted or required in the country of the seller. The currency code is required for second subitem also.

The total VAT amount can be optionally specified at the invoice position level (BT-X-327) also, but within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetGrandTotal Method

Return value: IAmount. [BT-112] [BT-X-330]

The total amount of the invoice with VAT (BT-112 = BT-109 + BT-110). This subitem can be recalculated using Calculate method. **It is required subitem within all standards and profiles at the invoice level.** This subitem can be optionally specified at the invoice position level (BT-X-330) also, but within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetRounding Method

Return value: IAmount. [BT-114]

The amount to be added to the invoice total to round the amount to be paid. This subitem can be optionally specified within all standards and profiles at the invoice level only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetTotalPrepaid Method

Return value: IAmount. [BT-113]

The sum of amounts which have been paid in advance. This amount is subtracted from the invoice total amount with VAT to calculate the amount due for payment. This subitem can be optionally specified within all standards and profiles at the invoice level only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetDuePayable Method

Return value: IAmount. [BT-115]

The outstanding amount that is requested to be paid (BT-115 = BT-112 - BT-113 + BT-114). The amount may be zero or negative (in that case the seller owes the amount to the buyer). This subitem can be recalculated using Calculate method. **It is required subitem within all standards and profiles at the invoice level.** The subitem is restricted at invoice position level.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

Calculate Method

Return value: sx_bool.

This metod helps you to calculate the totals - for the invoice as a whole or for the individual invoice position. You can call it using IMonetarySummation subitem from ISettlement after adding all invoice positions with prices and VAT information, also optionally charges and allowances at invoice and/or position level – this method fills all required amounts with actual values. The method calculates BT-131 or BT-106, BT-107, BT-108, BT-109, BT-110, BT-111, BT-112 and BT-115. It returns false if one or more required invoice elements are not filled or filled wrong.

subitems Parameter

Type: sx_bool.

When you call this method for the invoice as a whole, set “subitems” to true if you want recalculate all invoice positions before calculating of invoice totals. If you calculate invoice position totals, you can use this parameter for recalculation of subpositions if present.

IRefDoc Interface

This interface is the most widely used in the API invoice structure, it is used for reference description of many different documents related to this invoice, for example:

- a referenced sales order, issued by the seller
- a referenced purchase order, issued by the buyer
- a contarct reference
- the additional supporting documents substantiating the claims made in the invoice
- the preceding invoice references

As well as many other related documents. References to related documents can be used at invoice and position levels. The description of a link to a related document consists of several fields (properties), the most commonly used of which is the identifier or number of such a document (IssuerAssignedID). In most cases, this property is mandatory. Other properties are used depending on the context and type of the related document - for more details, please refer to the documentation of the e-invoice standard used. The additional supporting documents may contain not only a link, but also the document itself, embedded in the e-invoice - see more details for GetAttachment method of this interface.

It is a subitem of IAgreement, IDelivery, ISettlement and IAdvancePayment interfaces.

IssuerAssignedID Property

Type: sx_str. Access: full.

[BT-13] [BT-14] [BT-12] [BT-17] [BT-18] [BT-122] [BT-15] [BT-16] [BT-25] – invoice level, required

[BT-X-403] [BT-X-150] [BT-X-202] [BT-X-558] – invoice level, required, within extended profile of CII syntax only

[BT-128] – position level, required

[BT-X-27] – position level, required, within extended profile of CII syntax only

[BT-X-537] [BT-X-21] [BT-X-310] [BT-X-24] [BT-X-43] [BT-X-86] [BT-X-89] [BT-X-92] [BT-X-331] – position level, optional, within extended profile of CII syntax only

Identifier of therelated document, e.g. agreement or order number. Depending on the type of document and context of use, it is required or optional. It used at invoice and position level. For more details, please refer to the documentation of the e-invoice standard used.

URIID Property

Type: sx_str. Access: full. [BT-124] [BT-X-28]

External document location. The property can be used within all standards at invoice level for additional supporting documents only. It can be used at position level for additional supporting documents within extended profile of CII syntax also.

LineID Property

Type: sx_uint. Access: full.

[BT-132] [BT-X-538] [BT-X-311] [BT-X-25] [BT-X-29] [BT-X-44] [BT-X-87] [BT-X-90] [BT-X-93] [BT-X-540]

Use **LineIDStr** property for non-numeric values, type: sx_str.

Identifier of referenced position. Can be used at position level only.

TypeCode Property

Type: sx_uint. Access: full.

The code for identification of a document type. This property can be used within all standards and profiles in the cases of additional supporting documents (required property) and preceding invoice references.

RefTypeCode Property

Type: sx_str. Access: full.

The code for identification of scheme of the document type. This property can be used within all standards and profiles in the cases of additional supporting documents.

GetName Method

Return value: sx_str. [BT-123] [BT-X-299]

Gets the description of a relayed document. This property can be used within all standards and profiles in the case of additional supporting documents only.

“i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements. The method produces the exceptions if failed, see ErrorCode property of exception for details.

SetName Method

[BT-123] [BT-X-299]

Sets the description of a related document. This property can be used within all standards and profiles in the case of additional supporting documents only.

“s” parameter is new value of description. “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when new description should be added). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetAttachment Method

Return value: IAttachment. [BT-125] [BT-X-31]

An embedded related document. This subitem can be used within all standards and profiles in the case of additional supporting documents only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method

returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetIssueTime Method

Return value: IDateTime.

[BT-26] [BT-X-146] [BT-X-147] [BT-X-148] [BT-X- 149] [BT-X-151] [BT-X-200] [BT-X-201] [BT-X-203] [BT-X- 404] [BT-X-333] [BT-X- 556] [BT-X- 557] [BT-X-560] [BT-X-22] [BT-X-26] [BT-X-33] [BT-X-45] [BT-X-88] [BT-X-91] [BT-X-94] [BT-X-539]

Related document date. It can be used within extended profile of CII syntax only, excluding a case of preceding invoice reference at invoice level.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IAttachment Interface

An attached document embedded as binary object (Base64). The interface provides access to the content, MIME code and original filename of embedded document.

MimeCode Property

Type: sx_str. Access: full. [BT-125-1] [BT-X-31-1]

MIME code (file format) of the embedded document.

FileName Property

Type: sx_str. Access: full. [BT-125-2] [BT-X-31-2]

Original filename of the embedded document.

EncData Property

Type: sx_str. Access: full. [BT-125] [BT-X-31]

Binary content of attachment encoded using Base64 algorithm into a sequence of printable characters.

GetData Method

Stream Parameter

Type: ISequentialStream.

The method decodes value of **EncData** property from Base64 format and writes the binary content to

the stream.

SetData Method

Stream Parameter

Type: ISequentialStream.

The method encodes binary content from the stream to Base64 string and sets **EncData** property value.

IProcProject Interface

Detailed information on the associated project: project reference (number, etc.), project name.

ID Property

Type: sx_str. Access: full. [BT-11]

Project reference - number, etc.

Name Property

Type: sx_str. Access: full.

Project name.

IAccount Interface

A refernce that specifies where to book the relevant data into the buyer's financial accounts.

ID Property

Type: sx_str. Access: full.

A textual value that specifies where to book the relevant data into the buyer's financial accounts.

TypeCode Property

Type: sx_uint. Access: full.

A coded value: 1 - Financial, 2 - Subsidiary, 3 - Budget, 4 - Cost Accounting, 5 - Receivable, 6 - Payable, 7 - Job Cost Accounting. The property can be used within extended profile of CII syntax only.

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.

It is a subitem of IAgreement, IDelivery, ISettlement and IPaymentTerms interfaces.

ID Property

Type: sx_str. Access: full.

[BT-29] – optional, several are allowed (see **GetID/SetID** methods)

[BT-46] [BT-71] [BT-60] – optional

[BT-X-337] [BT-X-364] [BT-X-116] [BT-X-126] [BT-X-408] [BT-X-162] [BT-X-181] [BT-X-205] [BT-X-224] [BT-X-478] [BT-X-48] [BT-X-67] [BT-X-506] – optional, within extended profile of CII syntax only

Identification of the trade party. Use GetGlobalID method if you need to operate with specific identifier given to the trade party by a global registration authority.

Name Property

Type: sx_str. Access: full.

[BT-27] [BT-44] [BT-62] [BT-59] – required

[BT-70] – optional

[BT-X-335] [BT-X-362] [BT-X-128] [BT-X-406] [BT-X-207] [BT-X-226] [BT-X-476] [BT-X-504] – required if party is present, within extended profile of CII syntax only

[BT-X-164] [BT-X-183] [BT-X-50] [BT-X-69] – optional, within extended profile of CII syntax only

Name of the party, usually company name.

RoleCode Property

Type: role_code. Access: full.

[BT-X-543] [BT-X-544] [BT-X-545] [BT-X-546] [BT-X-547] [BT-X-548] [BT-X-549] [BT-X-550] [BT-X-551] [BT-X-552] [BT-X-553] [BT-X-554] [BT-X-468] [BT-X-483] [BT-X-541] [BT-X-542] [BT-X-511] – optional, within extended profile of CII syntax only

A code qualifying the role of the party according to the UNTDID 3035. The property can be used within extended profile of CII syntax only.

Description Property

Type: sx_str. Access: full.

Various legal description of the party, it is optional for seller party [BT-33] and for buyer party [BT-X-334] within extended profile of CII syntax only. For any other party it is restricted.

GetGlobalID Method

Return value: IGlobalID.

[BT-29] – optional, several are allowed

[BT-46] [BT-71] [BT-60] – optional, single (several are allowed within extended profile of CII syntax only)

[BT-X-338] [BT-X-365] [BT-X-117] [BT-X-127] [BT-X-409] [BT-X-163] [BT-X-182] [BT-X-206] [BT-X-225] [BT-X-479] [BT-X-49] [BT-X-68] [BT-X-507] – optional, several are allowed, within extended profile of CII syntax only

Specific identifier given to the trade party by a global registration authority - GLN, DUNS, BIC,ODETTE…

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetOrganization Method

The method returns IOrganization interface.

[BT-30] [BT-47] [BT-61] – optional

[BG-X-50] [BG-X-58] [BG-X-16] [BG-X-19] [BG-X-63] [BG-X-25] [BT-X-165] [BT-X-184] [BT-X-208] [BT-X-227] [BT-X-480] [BT-X-51] [BT-X-70] [BT-X-508] – optional, within extended profile of CII syntax only

Details about the trade party organization.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created).

GetContact Method

Return value: IContact.

[BG-6] [BG-9] – optional, single (several are allowed within extended profile of CII syntax only)

[BG-X-51] [BG-X-55] [BG-X-17] [BG-X-20] [BG-X-64] [BG-X-26] [BG-X-28] [BG-X-31] [BG-X-34] [BG-X-37]

[BG-X-39] [BG-X-74] [BG-X-8] [BG-X-11] [BG-X-78] – optional, within extended profile of CII syntax only

The trade party contact: contact person name, department name, phone, fax, email contact data and postal address.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created).

GetAddress Method

Return value: IAddress.

[BG-5] [BG-8] [BG-12] – required

[BG-15] – optional

[BG-X-52] [BG-X-56] [BG-X-21] [BG-X-65] [BG-X-29] [BG-X-32] [BG-X-35] [BG-X-38] [BG-X-40] [BG-X-75] [BG-X-9] [BG-X-12] [BG-X-79] – optional, within extended profile of CII syntax only

The trade party postal address: post code, address in three lines, city, country, country subdivision.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created).

GetUniversalURI Method

Return value: ICommCont.

[BT-34] [BT-49] – optional, **required for XRechnung invoices starting from verson 3.0 and higher**

[BT-X-341] [BT-X-368] [BT-X-125] [BT-X-143] [BT-X-412] [BT-X-160] [BT-X-179] [BT-X-198] [BT-X-222] [BT-X-241] [BT-X-256] [BT-X-482] [BT-X-65] [BT-X-83] [BT-X-510] – optional, within extended profile of CII syntax only

The electronic address of the party, **use scheme identificator “EM” as value of Format propery**.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created).

GetTaxID Method

Return value: ITaxID.

[BT-31] [BT-32] - optional

Seller VAT identifier (VAT number prefixed by a country code) or seller tax registration identifier (e.g. in some countries, if the seller is not registered as a tax paying entity then the buyer is required to withhold the amount of the tax and pay it on behalf of the seller).

[BT-48] - optional

Buyer VAT identifier (VAT number prefixed by a country code).

[BT-63] [BT-X-367] - required

Tax representative VAT identifier (VAT number prefixed by a country code).

[BT-X-340] [BT-X-144] [BT-X-411] [BT-X-161] [BT-X-180] [BT-X-199] [BT-X-223] [BT-X-242] [BT-X-257] [BT-X-481] [BT-X-66] [BT-X-84] [BT-X-509]

VAT identifier (VAT number prefixed by a country code).

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IOrganization Interface

Details about the organization: ID, name, postal address. It is a subitem of IParty interface.

Name Property

Type: sx_str. Access: full. [BT-28] [BT-45]

Trading business name of the party.

GetID Method

Return value: IOrgID. [BT-30] [BT-47] [BT-61]

Party identification - company registration number.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetAddress Method

Return value: IAddress. [BG-X-14] [BG-X-15] [BG-X-53] [BG-X-57] [BG-X-59] [BG-X-60] [BG-X-66] [BG-X-67] [BG-X-68] [BG-X-69] [BG-X-] [BG-X-70] [BG-X-71] [BG-X-72] [BG-X-76] [BG-X-80]

Contact postal address: post code, address in three lines, city, country, country subdivision.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

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. It is a subitem of IParty and IOrganization interfaces.

Postcode Property

Type: sx_uint. Access: full. [BT-38] [BT-53] [BT-67] [BT-78]

Use **PostcodeStr** property for non-numeric values, type: sx_str.

Post code of the contact.

Line1 Property

Type: sx_str. Access: full. [BT-35] [BT-50] [BT-64] [BT-75]

Address line 1 of the contact.

Line2 Property

Type: sx_str. Access: full. [BT-36] [BT-51] [BT-65] [BT-76]

Address line 2 of the contact.

Line3 Property

Type: sx_str. Access: full. [BT-162] [BT-163] [BT-164] [BT-75]

Address line 3 of the contact.

City Property

Type: sx_str. Access: full. [BT-37] [BT-52] [BT-66] [BT-77]

City of the contact.

Country Property

Type: **CountryCode2**. Access: full. [BT-40] [BT-55] [BT-69] [BT-80]

The code identifying the country of the contact. Country code according to the ISO 3166-1 (Alpha-2 code space). Use **ToCountryCode2** and **FromCountryCode2** methods of IInvoiceData for conversion between **CountryCode2** type and string value. The subitem can be used within all standards and profiles.

GetCountryPart Method

Return value: sx_str. [BT-39] [BT-54] [BT-68] [BT-79]

The country subdivision of the contact.

“i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements. The method produces the exceptions if failed, see ErrorCode property of exception for details.

SetCountryPart Method

The country subdivision of the contact.

“s” parameter is new value of description. The method produces the exceptions if failed, see ErrorCode property of exception for details.

IContact Interface

Detailed information about the trade party contact: contact person name, department name, phone, fax, email contact data and postal address. It is a subitem of IParty interface.

Person Property

Type: sx_str. Access: full. [BT-41] [BT-56]

Person name.

Department Property

Type: sx_str. Access: full.

Department name.

TypeCode Property

Type: cont_code. Access: full.

The code specifying the type of trade contact, to be chosen from the entries of UNTDID 3139. The subitem can be used within extended profile of CII syntax only.

GetTelephone Method

Return value: ICommCont. [BT-42] [BT-57]

Contact phone number.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method

returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetFax Method

Return value: ICommCont.

Contact fax number. The subitem can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetEmail Method

Return value: ICommCont. [BT-43] [BT-58]

Contact email address.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

ICommCont Interface

The contact information.

Number Property

Type: sx_str. Access: full.

The phone or fax number of the contact.

URI Property

Type: sx_str. Access: full.

The electronic address of the contact.

Format Property

Type: sx_str. Access: full.

The identification scheme identifier of the electronic address, it should be “**EM**” for electronic address. This identifier is allowed and required for the electronic address of the invoice parties only (not for contact emails).

IPaymentMeans Interface

A group of business terms providing information about the payment:

- The means, expressed as code, for how a payment is expected to be or has been settled
- 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)

It is subitem of ISettlement interface.

TypeCode Property

Type: sx_uint. Access: full. [BT-81]

The means, expressed as code, for how a payment is expected to be or has been settled, according to the UNTDID 4461. Distinction should be made between SEPA and non-SEPA payments, and between credit payments, direct debits, card payments and other instruments. **It is required property within all standards and profiles.**

Information Property

Type: sx_str. Access: full. [BT-82]

The means, expressed as text, for how a payment is expected to be or has been settled, such as cash, credit transfer, direct debit, credit card, etc.

GetCard Method

Return: IFinCard. [BG-18]

A group of business terms providing information about card used for payment contemporaneous with invoice issuance (only used if the buyer has opted to pay by using a payment card such as a credit or debit card). **An invoice shall contain maximum one payment card account and only if the type code [BT-81] is 48, 54 or 55.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetDebtorAccount Method

Return: IFinAccount. [BT-91]

The account to be debited by the direct debit. **An invoice shall contain maximum one payment mandate and only if the type code [BT-81] is 59.**

This is element of a group of business terms to specify a direct debit [**BG-19**], other elements are “mandate reference identifie” [**BT-89**] and “bank assigned creditor identifier” [**BT-90**].

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetCreditorAccount Method

Return: IFinAccount. [BG-17]

A group of business terms to specify credit transfer payments, elements [**BT-85**] and [**BT-84**], can be ised if the type code [**BT-81**] means SEPA credit transfer, local credit transfer or non-SEPA international credit transfer only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetCreditorInstitution Method

Return: IFinInst. [BT-86]

An identifier for the payment service provider where a payment account is located, such as a BIC or a national clearing code where required. No identification scheme to be used.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IFinCard Interface

A group of business terms providing information about card used for payment contemporaneous with invoice issuance (only used if the buyer has opted to pay by using a payment card such as a credit or debit card).

ID Property

Type: sx_str. Access: full. [BT-87]

The Primary Account Number (PAN) of the card used for payment.

In accordance with card payments security standards an invoice should never include a full card primary account number. At the moment PCI Security Standards Council has defined following: The first 6 digits and last 4 digits are the maximum number of digits to be shown.

Name Property

Type: sx_str. Access: full. [BT-88]

The name of the payment card holder.

IFinAccount Interface

A group of business terms to specify credit transfer payments.

IBAN Property

Type: sx_str. Access: full. [BT-84] [BT-91]

A unique identifier of the financial payment account, at a payment service provider, to which payment should be made [BT-84] or the account to be debited by the direct debit [BT-91].

Name Property

Type: sx_str. Access: full. [BT-85]

The name of the payment account, at a payment service provider, to which payment should be made.

ProprietaryID Property

Type: sx_str. Access: full.

National account number (not SEPA).

IFinInst Interface

An identifier for the payment service provider where a payment account is located, such as a BIC or a national clearing code where required. No identification scheme to be used.

BIC Property

Type: sx_str. Access: full.

BIC or a national clearing code.

IPaymentTerms Interface

Detailed information about payment terms.

Description Property

Type: sx_str. Access: full. [BT-20]

A textual description of the payment terms that apply to the amount due for payment (including description of possible penalties). In case the amount due for payment [**BT-115**] is positive, either the payment due date [**BT-9**] (see below) or this property shall be specified.

Information on cash discounts for prompt payment (Skonto) can be specified within this text (**pseudo-structured form**) for all standards and profiles or using GetPartialPaymentAmount, GetPenalty, GetDiscount methods (**structured form**) within extended profile of CII syntax only.

Pseudo-structured form: First segment "**SKONTO**", second segment amount of days ("**TAGE=N**"), third segment percentage ("**PROZENT=N**"). Percentage must be separated by dot with two decimal places. In case the base value of the invoiced amount is not provided in [**BT-115**] but as a partial amount, the base value shall be provided as fourth segment "**BASISBETRAG=N**" as semantic data type amount. Each entry shall start with a **#**, the segments must be separated by **#** and a row shall end with a **#**. A complete statement on cash discount for prompt payment shall end with a line break. All statements on cash discount for prompt payment shall be given in capital letters. Additional whitespaces (blanks, tabulators or line breaks) are not allowed. Other characters or texts than defined above are not allowed.

DirectDebitMandate Property

Type: sx_str. Access: full. [BT-89]

Unique identifier assigned by the payee for referencing the direct debit mandate. Used in order to pre-notify the buyer of a SEPA direct debit.

This is element of a group of business terms to specify a direct debit [**BG-19**], other elements are “bank assigned creditor identifier” [**BT-90**] and “debited account identifier” [**BT-91**].

GetTime Method

Return value: IDateTime.

Payment due date. [BT-9]

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetPartialPaymentAmount Method

Return value: IAmount. [BT-X-275]

Partial payment amount. This subitem can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetPenalty Method

Return value: IPaymentCorrection. [BG-X-43]

Detailed information about penalties. This subitem can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetDiscount Method

Return value: IPaymentCorrection. [BG-X-44]

Detailed information about payment discounts. This subitem can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetPayeeParty Method

Return value: IParty. [BG-X-77]

The information about the payee, i.e. the role that receives the payment, in case of multiple payees. The role of beneficiary may be filled by a party other than the seller, e.g. a factoring service. This group is only used when there are multiple beneficiaries (e.g. withholding tax or split payment). This subitem can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IPaymentCorrection Interface

Detailed information about payment discounts or penalties. Can be used within extended profile of CII syntax only.

Percent Property

Type: sx_double. Access: full. [BT-X-280] [BT-X-286]

Payment discount/penalty percentage.

GetBasisTime Method

Return value: IDateTime.

Maturity reference date. [BT-X-276] [BT-X-282]

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetBasisQuantity Method

Return value: IQuantity.

The period for the due date, e.g. as a number of days (15 days). [BT-X-277] [BT-X-283]

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetBasisAmount Method

Return value: IAmount. [BT-X-279] [BT-X-285]

Payment discount/penalty base amount.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetCorrectionAmount Method

Return value: IAmount. [BT-X-281] [BT-X-287]

Payment discount/penalty amount.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IDeliveryTerms Interface

Details of the delivery conditions. Can be used within extended profile of CII syntax only.

Code Property

Type: sx_str. Access: full. [BT-X-145]

The code specifying the type of delivery for these trade delivery terms. To be chosen from the entries in UNTDID 4053 + INCOTERMS List.

IChainEvent Interface

Information about the actual delivery time: date of delivery and achievement in terms of taxability.

GetTime Method

Return value: IDateTime.

Actual delivery time. **It is required subitem within all standards and profiles.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IProductChar Interface

The properties (attributes) of the goods and services invoiced.

TypeCode Property

Type: sx_str. Access: full. [BT-X-11]

Item property coded type. To ensure automated processing of the article attributes without bilateral reconciliation, only values from the code list UNTDED 6313+Factur-X-Extension should be used. The subitem can be used within extended profile of CII syntax only.

Description Property

Type: sx_str. Access: full. [BT-160]

Item attribute name. **It is required property within all standards and profiles.**

Value Property

Type: sx_str. Access: full. [BT-161]

Item attribute value. **It is required property within all standards and profiles.**

GetQuantity Method

Return value: IQuantity. [BT-X-12]

Item attribute value (numerical measurand). The subitem can be used within extended profile of CII syntax only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IProductClass Interface

A code for classifying the item by its type or nature.

Name Property

Type: sx_str. Access: full. [BT-X-13]

Name used to classify an item according to its type or nature. The property can be used within extended profile of CII syntax only.

GetCode Method

Return value: IMultiCode. [BT-158]

A code for classifying the item by its type or nature.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IMultiCode Interface

Code Property

Type: sx_str. Access: full.

A code for classifying the item by its type or nature. Classification codes are used to allow grouping of similar items for a various purpose e.g. public procurement (CPV), e-Commerce (UNSPSC) etc.

Format Property

Type: sx_str. Access: full.

The identification scheme identifier of the Item classification identifier in accordance with the entries of UNTDID 7143. **It is required property within all standards and profiles.**

Version Property

Type: sx_str. Access: full.

The version of the identification scheme.

IProductInstance Interface

Information about product instance. Can be used within extended profile of CII syntax only.

BatchID Property

Type: sx_str. Access: full. [BT-X-306]

Batch ID of the Item (trade product) instance.

SupplierSerialID Property

Type: sx_str. Access: full. [BT-X-307]

Serial ID of the Item (trade product) instance.

ICurrencyExchange Interface

Specification of currency exchange rate for a certain exchange date. Can be used within extended profile of CII syntax only.

Source Property

Type: **CurrCode3**. Access: full. [BT-X-258]

Invoice currency code according to the ISO 4217 (Alpha-3 code space). **It is required property.** Use **ToCurrCode3** and **FromCurrCode3** methods of IInvoiceData for conversion between **CurrCode3** type and string value.

Target Property

Type: **CurrCode3**. Access: full. [BT-X-259]

Local currency code according to the ISO 4217 (Alpha-3 code space). **It is required property.** Use **ToCurrCode3** and **FromCurrCode3** methods of IInvoiceData for conversion between **CurrCode3** type and string value.

Rate Property

Type: sx_double. Access: full. [BT-X-260]

The currency exchange rate. **It is required property.**

GetTime Method

Return value: IDateTime interface.

The currency exchange date and time. [BT-X-261]

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IAdvancePayment Interface

Tax information on advance payments received. Can be used within extended profile of CII syntax only.

GetAmount Method

Return value: IAmount. [BT-X-291]

Amount to be paid in advance. **It is required subitem.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetTime Method

Return value: IDateTime.

Date of advanced payment. [BT-X-292]

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetTax Method

Return value: ITax. [BT-X-46]

Tax information on advanced payments. **At least one subitem is required.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetReferencedDoc Method

Return value: IRefDoc. [BT-X-85]

Preceding invoice reference for advance payment. The individual invoide shall be stated so that combined payments need to be split per invoice. To be used in case:

- preceding partial invoices are refered to from a final invoice
- preceding pre-payment invoices are refered to from a final invoice

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IPrepaidPayment Interface

A group of information elements that contain information about third-party claims. Note: third-party claims only affect the amount to be paid, but are not part of the actual invoice. **The subitem can be used within Extension XRechnung of UBL syntax only.**

ID Property

Type: sx_str. Access: readonly. [BT-DEX-001]

The type of third-party claim. **It is required property.**

InstructionID Property

Type: sx_str. Access: readonly. [BT-DEX-002]

A clear description of the third-party receivable within the invoice. Note: This information element is used to clearly distinguish between several "THIRD-PARTY PAYMENT" [BG-DEX-09] information elements. **It is required property.**

GetAmount Method

Return value: IAmount. [BT-DEX-003]

The amount of the third-party receivable. Note: amounts from third-party receivables are gross amounts. No VAT breakdown is provided. **It is required subitem.**

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IChainConsignment Interface

Detailed information on consignment or shipment. Should be used in case of different goods or services recipient as the buyer. Can be used within extended profile of CII syntax only.

GetTransportMovement Method

Return value: ITransportMovement. [BT-X-152]

Logistics transport movement.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). “i” parameter is index of an element. Can be greater than or equal to 0 and less than number of elements (or equal, when c parameter is true). The method produces the exceptions if failed, see ErrorCode property of exception for details.

ITransportMovement Interface

Specified logistics transport movement: maritime, rail, road, air, etc.

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.

Can be used within extended profile of CII syntax only.

Code Property

Type: sx_str. Access: full.

The code specifying the mode, such as air, sea, rail, road or inland waterway, for this logistics transport movement:

- 0 - ransport mode not specified
- 1 - maritime transport
- 2 - rail transport
- 3 - road transport
- 4 - air transport
- 5 - mail
- 6 - multimodal transport
- 7 - fixed transport installation
- 8 - inland water transport
- 9 - transport mode not applicable

INote Interface

Detailed information as a code and a free text to a whole invoice or to its position. Bilateral agreed text items in form of a code or a free text. It is subitem of IInvoiceDescription and ILineDoc interfaces.

Content Property

Type: sx_str. Access: full. [BT-22] [BT-127]

A free text that gives unstructured information that is relevant to the invoice as a whole or to the invoice position.

SubjectCode Property

Type: sx_str. Access: full. [BT-21] [BT-X-10]

Code for qualifying the free text of the note according to the UNTDID 4451. The property can be used on the invoice level within all standards and profiles and at invoice position level within extended profile of CII syntax.

ContentCode Property

Type: sx_str. Access: full. [BT-X-5] [BT-X-9]

Bilaterally agreed free text of the note which here is being transmitted as a code, subset of UNTDID 4451: REG, AAK, AAJ, PMT. It can be used within extended profile of CII syntax only.

ITimePeriod Interface

Detailed information on the invoicing period: description, invoicing period start and end dates. It is subitem of IInvoiceDescription and ILineSettlement interfaces.

Description Property

Type: sx_str. Access: full. [BT-X-264]

Invoicing period description (free text). Can be used in context of [**BG-14**] and within extended profile of CII syntax only.

GetStart Method

Return value: IDateTime. [BT-73] [BT-134]

Start date of the period. This subitem is restricted in context of [**BT-X-6**].

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetEnd Method

Return value: IDateTime. [BT-74] [BT-135]

End date of the period. This subitem is restricted in context of [**BT-X-6**].

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

GetComplete Method

Return value: IDateTime.

Contractual due date of the invoice. This subitem is allowed in context of [**BT-X-6**] only.

If “c” parameter is false, the method returns existing element or null. If parameter is true, the method returns element (existing or just created). The method produces the exceptions if failed, see ErrorCode property of exception for details.

IAmount Interface

The goods value using in different invoice parts: basic amount, tax amount, actual amount, allowance and charge amount, total amount.

Important! Please note, when you use trial license, all amount values are randomly multiplied by a value from 2 to 9. This is done for opened and created invoices.

Currency Property

Type: **CurrCode3**. Access: full.

The currency code according to the ISO 4217 (Alpha-3 code space). Use **ToCurrCode3** and **FromCurrCode3** methods of IInvoiceData for conversion between **CurrCode3** type and string value.

Amount Property

Type: sx_double. Access: full.

The amount in accounting currency or in currency specified by **Currency** property. **It is required property.**

IQuantity Interface

Information about billed, free, packaged quantity. The quantity 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 = qubic meter, KGM = kilogram, MTR = metre, PCE = piece, TNE = tone.

Quantity Property

Type: sx_double. Access: full.

The quantity value in units specified by **Unit** property. **It is required property.**

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
- datetime_format_CCYYMMDDHHMM (Order-X only)
- datetime_format_CCYYMMDDHHMMSS (Order-X only)

SetDate Method

The method changes date value and/or format.

“f” parameter (**datetime_format)** - date format.

“tm” parameter (sx_time) - date value.

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

Scheme of a global indentification.

Type: **global_id**. Access: readonly.

SetID Method

Identification of a party or product.

“f” parameter (**global_id**) - type of a global indentification.

“id” parameter (sx_str) - identification of a party or product.

IOrgID Interface

Identification of 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 (**org_id**) - type of an indentification.

“id” parameter (sx_str) - identification of the organization.

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 (**global_id**) - type of the tax ID.

“id” parameter (sx_str) - tax or VAT number.

IOrderX Interface

The interface provides functionality for processing XML based electronic orders that are Order-X conform.

You can create a new instance of this object using the IInvoiceDocument::CreateOrderX method.

C++ We highly recommend you use an instance of an ObjPtr template class where you can use an EInvoice::IInvoiceData pointer in a safe manner.

Standard Property

Type: einv_standard. Access: readonly.

The property retrieves standard (version) of e-order.

Profile Property

Type: einv_profile. Access: readonly.

The property retrieves profile of e-order - EN16931 (COMFORT) or EXTENDED.

Test Property

Type: sx_bool. Access: full.

The property defines an e-order as test order. It can be set within extended profile.

BusinessProcess Property

Type: sx_str. Access: full.

The property defines the business process identifier of e-order. It can be set within all profiles.

Description Property

Type: IOrderDescription. Access: readonly.

The property retrieves interface which provides description properties of e-order – general properties for the whole document such as document number, document title, type code, issue date, notes, etc.

Transaction Property

Type: IOrderTransaction. Access: readonly.

The property retrieves interface which represents main parts of e-order – structured information about all transactions in a document: individual e-order positions, information about goods and services,

prices, reference products, etc.

Open Method

The method loads e-order from XML stream or file. Method returns validation status of loaded invoice. This method allows to load partially filled e-orders for completing of filling or extracting filled data (use einv_options_def option) or can check all required elements (use einv_options_check_xml option).

Stream Parameter

Type: ISequentialStream.

The invoice data stream. Please reset stream before load.

Opt Parameter

Type: sx_flags.

It can be einv_options_def or einv_options_check_xml.

Create Method

The method creates new empty e-order. Fill it with complete data, then call **Save** method.

St Parameter

Type: einv_standard.

The parameter should be “einv_standard_orderx”.

Pr Parameter

Type: einv_profile.

The parameter specifies the necessary profile of e-order.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

Save Method

The method saves an e-order to the XML stream or file. You can check completeness of e-oder by calling **CheckRequirements** method before saving.

Stream Parameter

Type: ISequentialStream.

The stream for saving of XML data.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

CheckRequirements Method

The method checks completeness of e-order before saving. It returns false, if it is non-. More detailed information about missed elements you can get using callback interface **IInvoiceValidation**. 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.

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: IInvoiceValidation.

The callback handler. The IInvoiceValidation interface must implemented by client application.

IOrderDescription Interface

The interface provides description properties of e-order – general properties for the whole document such as document number, document title, type code, issue date, notes, etc.

IOrderTransaction Interface

The interface represents main parts of e-order – structured information about all transactions in a document: individual e-order positions, information about goods and services, prices, reference products, etc.

IOrderBIS Interface

The interface provides functionality for processing XML based electronic orders that are Peppol BIS conform.

You can create a new instance of this object using the IInvoiceDocument::CreateOrderBIS method.

C++ We highly recommend you use an instance of an ObjPtr template class where you can use an EInvoice::IInvoiceData pointer in a safe manner.

Standard Property

Type: einv_standard. Access: readonly.

The property retrieves standard (version) of e-order.

Namespace Property

Type: einv_bis. Access: readonly.

The property retrieves namespace of Peppol BIS order (method of order use).

Transaction Property

Type: ITransactionBIS. Access: readonly.

The property retrieves interface which provides description properties and represents main parts of e-order – such as document number, document title, type code, issue date, notes, individual e-oder positions, information about goods and services, prices, reference products, etc.

Open Method

The method loads e-order from XML stream or file. Method returns validation status of loaded invoice. This method allows to load partially filled e-orders for completing of filling or extracting filled data (use einv_options_def option) or can check all required elements (use einv_options_check_xml option).

Stream Parameter

Type: ISequentialStream.

The invoice data stream. Please reset stream before load.

Opt Parameter

Type: sx_flags.

It can be einv_options_def or einv_options_check_xml.

Create Method

The method creates new empty e-order. Fill it with complete data, then call **Save** method.

St Parameter

Type: einv_standard.

The parameter should be “einv_standard_order_bis”.

Ns Parameter

Type: einv_bis.

The parameter is namespace of Peppol BIS order (method of order use).

Spec Parameter

Type: sx_str.

The parameter is specification identification of Peppol BIS order.

BProc Parameter

Type: sx_str.

The parameter is business process type identifier of Peppol BIS order.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

Save Method

The method saves an e-oder to the XML stream or file. You can check completeness of e-oder by calling **CheckRequirements** method before saving.

Stream Parameter

Type: ISequentialStream.

The stream for saving of XML data.

Errors

The method produces the exceptions if failed, see ErrorCode property of exception for details.

CheckRequirements Method

The method checks completeness of e-order before saving. It returns false, if it is non-. More detailed information about missed elements you can get using callback interface **IInvoiceValidation**. 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.

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: IInvoiceValidation.

The callback handler. The IInvoiceValidation interface must implemented by client application.

ITransactionBIS Interface

The interface provides description properties and represents main parts of e-order – such as document number, document title, type code, issue date, notes, individual e-oder positions, information about goods and services, prices, reference products, etc.

Specification Property

Type: sx_str. Access: readonly.

The property defines the specification identification of Peppol BIS order.

BusinessProcess Property

Type: sx_str. Access: readonly.

The property defines the business process type identifier of Peppol BIS order.

ID Property

Type: sx_str. Access: full.

A number to uniquely identify the order 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.

Enumerations

einv_standard

The enumeration defines supported standards of invoices.

einv_standard_unknown – standard is undefined

einv_standard_zugferd – ZUGFeRD 1.0

einv_standard_zugferd20 – ZUGFeRD 2.0

einv_standard_facturx – FACTUR-X 1.0 (ZUGFeRD 2.1/2.2/2.3)

einv_standard_xrechnung – XRechnung 1.2, CII syntax

einv_standard_xrechnung20 – XRechnung 2.0, CII syntax

einv_standard_xrechnung21 – XRechnung 2.1, CII syntax

einv_standard_xrechnung22 – XRechnung 2.2, CII syntax

einv_standard_xrechnung23 – XRechnung 2.3.1, CII syntax

einv_standard_xrechnung30 – XRechnung 3.0.1, CII syntax

einv_standard_cii_family – CII syntax

einv_standard_ubl_xr – XRechnung 1.2, UBL syntax

einv_standard_ubl_xr20 – XRechnung 2.0, UBL syntax

einv_standard_ubl_xr21 – XRechnung 2.1, UBL syntax

einv_standard_ubl_xr22 – XRechnung 2.2, UBL syntax

einv_standard_ubl_xr23 – XRechnung 2.3.1, UBL syntax

einv_standard_ubl_xr30 – XRechnung 3.0.1, UBL syntax

einv_standard_ubl_cn30 – XRechnung 3.0.1 (CreditNote), UBL syntax

einv_standard_ubl_bis30 – Peppol BIS Billing 3.0

einv_standard_ubl_biscn30 – Peppol BIS Billing 3.0 (CreditNote)

einv_standard_ubl_family – UBL syntax

einv_standard_ebinterface60 – ebInterface 6.0 (https://www.erechnung.gv.at/erb)

einv_standard_ebinterface61 – ebInterface 6.1 (https://www.erechnung.gv.at/erb)

einv_standard_fatturapa12 – FatturaPA 1.2.2 (https://www.fatturapa.gov.it/en/lafatturapa)

einv_profile

The enumeration defines profiles of invoices.

einv_profile_unknown – profile is undefined

einv_profile_minimum – MINIMUM

einv_profile_basic_wl – BASIC WL

einv_profile_basic – BASIC

einv_profile_EN16931 – EN16931 conform (COMFORT for ZUGFeRD 1.0)

einv_profile_extended – EXTENDED (or Extension for XRechnung)

einv_bis

The enumeration defines allowed namespaces of Peppol BIS oder.

einv_bis_unknown – namespace is undefined

einv_bis_order - urn:oasis:names:specification:ubl:schema:xsd:Order-2

einv_bis_order_response - urn:oasis:names:specification:ubl:schema:xsd:OrderResponse-2

einv_bis_order_change - urn:oasis:names:specification:ubl:schema:xsd:OrderChange-2

einv_bis_order_cancelation - urn:oasis:names:specification:ubl:schema:xsd:OrderCancellation-2

einv_bis_catalogue - urn:oasis:names:specification:ubl:schema:xsd:Catalogue-2

einv_bis_application_response -urn:oasis:names:specification:ubl:schema:xsd:ApplicationResponse-2

einv_bis_despatch_advice - urn:oasis:names:specification:ubl:schema:xsd:DespatchAdvice-2

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_facturx

For invoice creating only, this option declares FACTUR-X standard.

einv_options_xrechnung21

For invoice creating only, this option declares FACTUR-X standard with embedded XRechnung 2.1 invoice.

einv_options_xrechnung22

For invoice creating only, this option declares FACTUR-X standard with embedded XRechnung 2.2 invoice.

einv_options_xrechnung23

For invoice creating only, this option declares FACTUR-X standard with embedded XRechnung 2.3 invoice.

einv_options_xrechnung30

For invoice creating only, this option declares FACTUR-X standard with embedded XRechnung 3.0 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_options_allow_ubl

Allow import or export of invoices using UBL syntax.

einv_options_ubl_codes

Adjust the used codes between UN/CEFACT CII and UBL standards automatically, for example BT-8 values (UNTDID2475/UNTDID2005).

einv_status

The enumeration defines validation status for opening of invoice file.

einv_status_not_pdf

The file is not a PDF (for opening PDF/A based invoices).

einv_status_enc_pdf

PDF is encrypted, this is not allowed for invoices (for opening PDF/A based invoices).

einv_status_not_pdfa

PDF is not a PDF/A (for opening PDF/A based invoices).

einv_status_no_meta

PDF has no metadata (for opening PDF/A based invoices).

einv_status_err_meta

PDF has wrong metadata (for opening PDF/A based invoices).

einv_status_no_emb

PDF has no embedded invoice (for opening PDF/A based invoices).

einv_status_err_emb

PDF has problems with embedded invoice (for opening PDF/A based invoices).

einv_status_nonconform

PDF/A is non-conform (for opening PDF/A based invoices).

einv_status_autogen

PDF contains visualization of XML invoice, it is not e-invoice (for opening PDF/A based invoices).

einv_status_not_xml

The file is not a XML (for opening XML based invoices).

einv_status_err_xml_ns

Requared e-invoice/e-order namespaces are missing.

einv_status_err_xml_root

Wrong root name.

einv_status_err_unk_prof

Invoice profile is not defined or wrong.

einv_status_err_xml_inv

XML structure is wrong.

einv_status_ubl_syntax

The e-incoice is in UBL syntax.

einv_status_valid

The valid e-invoice/e-order.

einv_imp_stat

The enumeration defines import status for opening of invoice file.

einv_imp_stat_not_xml

The file is not an XML.

einv_imp_stat_wrong_xml

XML structure is wrong.

einv_imp_stat_std_not_auth

Invoice format is not permitted by SDK license.

einv_imp_stat_std_cii

The valid e-invoice, UN/CEFACT CII syntax.

einv_imp_stat_std_ubl

The valid e-invoice, UBL syntax.

einv_imp_stat_std_import

The e-invoice in national format.

einv_imp_stat_ord_cii

The valid Order-X.

einv_imp_stat_ord_bis

The valid Peppol BIS Order.

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.

PDF Xpansion Control

This section contains the detailed description of PDF Xpansion SDK control objects for different environments implemented in PDF Xpansion SDK.

Application Property

Type: IApplication. Access: readonly.

This property provides a root of the PDF Xpansion SDK API and it is an alternative way to get an entry point of SDK. Please note that this property can be null if not all necessary redistributable files of SDK are available or correctly registered. You can't use control in this case, all properties and methods will generate the errors. Do not forget - PDF Xpansion SDK need to be authorized before you can use it, see more information here.

Document Property

Type: IViewableDocument. Access: full.

The property represents a document displayed in the control. Set this property to NULL for unloading of loaded document.

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: IViewerConfig. 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 activate any tool mode you need to set appropriate event handler.

KeysScroll Property

Type: sx_bool. Access: full.

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

WheelScroll Property

Type: sx_bool. Access: full.

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

WheelZoom Property

Type: sx_bool. Access: full.

The property enables processing of mouse wheel events (with pressed Ctrl key) for scaling of document in the viewer.

GestureScroll Property

Type: sx_bool. Access: full.

The property enables processing of touch input gestures for scrolling of document in the viewer.

GestureZoom Property

Type: sx_bool. Access: full.

The property enables processing of touch input gestures for scaling of document in the viewer.

GestureRotate Property

Type: sx_bool. Access: full.

The property enables processing of touch input gestures for changing of page orientation in the viewer.

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.

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.

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 IViewer 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

C++ 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 completelly drawed 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 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 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: 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: IViewerConfig. 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 completelly 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 IViewerConfig::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 completelly) 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 cleares history of the viewer operations (scrolling, navigation, relayout, 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 rectange 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 ractangle 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).

C++ 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

C++ 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**.

C++ 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.

.NET: 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.

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-coodinate of pointer in the viewport coordinate system.

Y Property

Type: sx_uint. Access: readonly.

The property value retrieves y-coodinate 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 IViewerConfig:: **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 cannot 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 caneled

Page Property

Type: sx_uint. Access: readonly.

The property value retrieves an index of page where tool is loacted 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 completelly 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 annoations, 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, polygone).

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 redrawed.

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::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 of 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 profides 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 interfaces represents a metadata object implemented by SDK. You can create new instance of document object using IAppFactory interface.

C++ 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::XHTML Namespace

This namespace contains some interfaces for processing documents and files in HTML and/or SVG formats, more precisely, to import such content into the PDF documents.

SX::Redesign Namespace

The mass creation of PDF documents based on templates makes it possible to efficiently create a large number of personalized documents. A standardized template is used that contains elements such as letterhead, header and footer to ensure a consistent corporate identity. These templates can then be filled with individual customer data to create customized documents such as invoices or quotes.

This namespace contains some interfaces which implement the possibility of mass creation of PDF documents.

The templates can either be created “on the fly” using interfaces in this namespace or designed using the interactive editor in our consumer product **Perfect Print 12**. The advantage lies in the time saved and the avoidance of manual errors, as all documents are generated automatically and uniformly.

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.

C++ We highly recommend you use an instance of an ObjPtr template class where you can use a **Content::IRichContent** pointer in a safe manner.

.NET:

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 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 than 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 a 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 dimenshions of object (position is irrelevant in this case) or use matrix returned by **GetTransformation** method to get boundinx 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.

TECHNICAL SUPPORT

Please use our ticket system for any requests or questions.