# Table of Contents

**Part I novaPDF SDK**

1. **Introduction** .......................................................... 11
2. **Overview** .............................................................. 11
   - Installation ............................................................... 11
   - System requirements .............................................. 12
   - Components .......................................................... 12
   - Network use ........................................................... 14
   - Multiple printers ................................................... 15
3. **Integration** ......................................................... 16
   - How to integrate .................................................... 16
   - How to make the release build ............................... 16
   - Customize your setup ............................................. 18
4. **novaPDF COM** .................................................... 27
   - How to register COM ............................................. 27
   - How to use the COM .............................................. 28
   - How to set printer options ..................................... 29
   - Private and public profiles ..................................... 29
   - How to register for messages ................................. 29
   - How to use events ................................................. 30
   - Multithreading applications ................................. 30
   - Use temporary printers ....................................... 31
   - Working with Layout ............................................ 32
   - Working with Actions .......................................... 32
5. **Reference** ............................................................ 33
   - Profile option strings .......................................... 33
   - Windows messages ............................................. 63
   - What is INovaPdfOptions ....................................... 64
   - INovaPdfOptions ................................................ 70
   - AddAction ............................................................ 70
   - AddAction2 .......................................................... 71
   - AddBookmarkDefinition ....................................... 72
   - AddBookmarkDefinition2 ....................................... 73
   - AddCustomProperty ............................................ 74
AddCustomProperty
AddNovaPrinter
AddNovaPrinterExt
AddNovaPrinterExt2
AddOverlay
AddOverlay2
AddProfile
AddProfile2
AddSaveRule
AddSaveRule2
AddUserTag
AddUserTag2
AddWatermarkImage
AddWatermarkImage2
AddWatermarkText
AddWatermarkText2
ClearCustomProperties
ClearFontOptions
ClearSaveRules
ClearUserTags
ConvertExcelDocument
ConvertExcelDocument2
ConvertPowerPointDocument
ConvertPowerPointDocument2
ConvertPublisherDocument
ConvertPublisherDocument2
ConvertVisioDocument
ConvertVisioDocument2
ConvertWordDocument
ConvertWordDocument2
CopyProfile
CopyProfile2
DeleteAction
DeleteAction2
DeleteBookmarkDefinition
DeleteCustomProperty
DeleteLayout
DeleteLayout2
DeleteNovaPrinter
DeleteNovaPrinter2
DeleteOverlay
DeleteOverlay2
DeleteOverrideOptions
DeletePrinterActivePublicProfile
DeletePrinterActivePublicProfile2
DeleteProfile
DeleteProfile2
DeleteSaveRule
DeleteUserTag
DeleteWatermarkImage
DeleteWatermarkImage2
DeleteWatermarkText
DeleteWatermarkText2
DisableAction .......................................................... 104
DisableAction2 ....................................................... 105
DisableActionType .................................................. 105
DisableCustomProperty ........................................... 105
DisableSaveRule ..................................................... 106
DisableUserTag ...................................................... 106
EnableAction .......................................................... 107
EnableAction2 ........................................................ 107
EnableActionType .................................................. 107
EnableCustomProperty .......................................... 108
EnableSaveRule ..................................................... 108
EnableUserTag ...................................................... 108
GetAction .............................................................. 109
GetAction2 ............................................................. 109
GetActionCount ...................................................... 110
GetActionOptionBool ............................................ 110
GetActionOptionBool2 .......................................... 111
GetActionOptionEncryptedString ......................... 111
GetActionOptionEncryptedString2 ....................... 112
GetActionOptionFloat ........................................... 113
GetActionOptionFloat2 ......................................... 113
GetActionOptionLong ........................................... 114
GetActionOptionLong2 ......................................... 115
GetActionOptionString ......................................... 115
GetActionOptionString2 ....................................... 116
GetActiveProfile .................................................... 116
GetActiveProfile2 ............................................... 117
GetBookmarkDefinition ........................................ 117
GetBookmarkDefinition2 ...................................... 118
GetBookmarkDefinitionCount ................................ 119
GetContentLayout ................................................ 120
GetContentLayout2 ................................................. 120
GetCustomProperty .............................................. 121
GetCustomProperty2 ............................................. 121
GetCustomPropertiesCount .................................. 122
GetFirstProfile ..................................................... 122
GetFirstProfile2 ................................................... 123
GetFontOption ...................................................... 123
GetFontOption2 .................................................... 124
GetLayout ............................................................. 124
GetLayout2 .......................................................... 125
GetLayoutCount .................................................... 125
GetLayoutCount2 .................................................. 126
GetLayoutOptionBool ........................................... 126
GetLayoutOptionBool2 ......................................... 127
GetLayoutOptionFloat ......................................... 127
GetLayoutOptionFloat2 ....................................... 128
GetLayoutOptionLong ......................................... 128
GetLayoutOptionLong2 ....................................... 129
GetLayoutOptionString ....................................... 130
GetLayoutOptionString2 ..................................... 131
GetNextProfile ..................................................... 131
GetNextProfile2 ................................................... 132
GetOptionBool ...................................................... 132
GetOptionEncryptedString
GetOptionEncryptedString2
GetOptionLong
GetOptionString
GetOptionString2
GetOverlay
GetOverlay2
GetOverlayCount
GetOverlayOptionBool
GetOverlayOptionBool2
GetOverlayOptionEncryptedString
GetOverlayOptionEncryptedString2
GetOverlayOptionFloat
GetOverlayOptionFloat2
GetOverlayOptionLong
GetOverlayOptionLong2
GetOverlayOptionString
GetOverlayOptionString2
GetOverrideOptionBool
GetOverrideOptionBool2
GetOverrideOptionEncryptedString
GetOverrideOptionEncryptedString2
GetOverrideOptionLong
GetOverrideOptionLong2
GetOverrideOptionString
GetOverrideOptionString2
GetPDFFileName
GetPDFFileName2
GetSaveRule
GetSaveRule2
GetSaveRulesCount
GetSignature
GetSignature2
GetUserTag
GetUserTag2
GetUserTagsCount
GetWatermarkImage
GetWatermarkImage2
GetWatermarkImageCount
GetWatermarkImageOptionBool
GetWatermarkImageOptionBool2
GetWatermarkImageOptionEncryptedString
GetWatermarkImageOptionEncryptedString2
GetWatermarkImageOptionFloat
GetWatermarkImageOptionFloat2
GetWatermarkImageOptionLong
GetWatermarkImageOptionLong2
GetWatermarkImageOptionString
GetWatermarkImageOptionString2
GetWatermarkTextOptionBool
GetWatermarkTextOptionBool2
GetWatermarkTextOptionFloat
GetWatermarkTextOptionFloat2
GetWatermarkTextOptionLong
GetWatermarkTextOptionLong2
GetWatermarkTextOptionString
5 **Samples** .......................................................... 215

What sample to choose ........................................ 215

Access .................................................................... 216

PDF Reports .......................................................... 216

**ASP.NET** .......................................................... 218

Hello World ......................................................... 218

**Delphi** ............................................................. 221
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCL Converter</td>
<td>221</td>
</tr>
<tr>
<td>Hello World Delphi</td>
<td>225</td>
</tr>
<tr>
<td>Convert Office Docs</td>
<td>228</td>
</tr>
<tr>
<td>Word OLE Delphi</td>
<td>231</td>
</tr>
<tr>
<td>Override Options</td>
<td>233</td>
</tr>
<tr>
<td><strong>C#</strong></td>
<td></td>
</tr>
<tr>
<td>Hello World CSharp</td>
<td>236</td>
</tr>
<tr>
<td>CSharp Converter</td>
<td>239</td>
</tr>
<tr>
<td>Word OLE CSharp</td>
<td>241</td>
</tr>
<tr>
<td>Convert Office Docs</td>
<td>244</td>
</tr>
<tr>
<td>Override Options</td>
<td>246</td>
</tr>
<tr>
<td><strong>C++</strong></td>
<td></td>
</tr>
<tr>
<td>Hello World</td>
<td>250</td>
</tr>
<tr>
<td>Hello World (network)</td>
<td>254</td>
</tr>
<tr>
<td>Convert Office Docs</td>
<td>255</td>
</tr>
<tr>
<td>MFC Converter</td>
<td>258</td>
</tr>
<tr>
<td>MFC Scribble</td>
<td>262</td>
</tr>
<tr>
<td>Temporary printer</td>
<td>264</td>
</tr>
<tr>
<td>Multiple printers</td>
<td>267</td>
</tr>
<tr>
<td>Override Options</td>
<td>271</td>
</tr>
<tr>
<td><strong>Java</strong></td>
<td></td>
</tr>
<tr>
<td>Hello World</td>
<td>275</td>
</tr>
<tr>
<td>Word OLE</td>
<td>275</td>
</tr>
<tr>
<td><strong>VBNNet</strong></td>
<td></td>
</tr>
<tr>
<td>Hello World VBNNet</td>
<td>286</td>
</tr>
<tr>
<td>VBNNet Converter</td>
<td>288</td>
</tr>
<tr>
<td>Word OLE VBNNet</td>
<td>290</td>
</tr>
<tr>
<td>Convert Office Docs</td>
<td>291</td>
</tr>
<tr>
<td>Override Options</td>
<td>294</td>
</tr>
<tr>
<td><strong>Install</strong></td>
<td></td>
</tr>
<tr>
<td>Installation package bundle</td>
<td>298</td>
</tr>
</tbody>
</table>
novaPDF SDK

Part I
1 novaPDF SDK

1.1 Introduction

novaPDF SDK is a PDF software development kit that programmers can use to add the ability to create PDF files in their own applications. As a COM Object, novaPDF SDK can be integrated in any Windows 2008 Server R2/Vista/Windows 7/Windows 8/Windows 2012 Server/Windows 10/Windows 2016 Server application programmed in a language that supports COM Objects (C/C++/C#, Visual C, Delphi, Visual Basic ...).

novaPDF SDK includes:
- a COM interface for customizing novaPDF printer options.
- novaPDF SDK documentation including several code samples (ASP.NET, C#, C++, Delphi, Java, Ms Access, Visual Basic, VBNet)
- novaPDF SDK printer to distribute (royalty free licensing)

Details regarding novaPDF SDK:
- It is mandatory to distribute the novaPDF SDK printer in your application's setup. This is done under royalty free licensing (you only pay for the SDK license). If you don't want to show novaPDF SDK under the list of Printers, you can use a temporary printer each time your application creates a PDF file (meaning the client won't see novaPDF in the list of printers)
- You can integrate it without ordering, and purchase a license only after you have fully tested it.
- The licensed novaPDF SDK (COM object) lets users create PDFs without the watermark only from your application, not by printing directly to the printer. Registering the COM object does not register the novaPDF SDK printer.
- If you purchase a license you get free priority support.

Restrictions
You are not allowed to create a PDF printer driver using novaPDF SDK, or another application similar to a PDF printer driver (whose main purpose is to create PDF files). You can integrate and distribute novaPDF SDK with your application, as long as your application does some mandatory pre-processing operations to the resulting PDF files.

1.2 Overview

1.2.1 Installation

Install
To install novaPDF SDK on a computer you need to have administrative rights. The installation process does not take much time. All you need to do is to follow the instructions of the "novaPDF 11 SDK Developer Setup" wizard.

There is no need to reboot at the end of the setup; you can run the program right after it is installed on your machine. If you have already installed an older version of novaPDF SDK, you can install the new version on top of the older one, without uninstalling it. Your existing option profiles will be preserved when installing a new version.
1.2.2 System requirements

To install novaPDF 11 SDK Developer you need one of the following operating systems:

- Windows 2016 Server
- Windows 10
- Windows 2012 Server
- Windows 8.1
- Windows 8
- Windows 7
- Windows 2008 Server R2
- Windows Vista

novaPDF 11 SDK Developer requires Microsoft .NET Framework 4.6. If not already installed, it will be downloaded and installed by novaPDF 11 SDK Developer setup.

It needs approximately 375MB of free space.

1.2.3 Components

novaPDF 11 SDK Developer installs files in two folders:

In "C:\Program Files\Softland\novaPDF 11" the installer will create the following folder structure:

Driver
Contains printer driver files and some tools, like Printer Manager

Tools
Contains Profile Manager files

Server
Contains the novaPDF Server service files that works with the profile database

SDK\Doc
Contains the help files and the license files

SDK\Installer
Contains the msi files that will be injected with your application and license information to be distributed with your application.
- novaPDF11PrinterDriver(x64).msi, novaPDF11PrinterDriver(x86).msi - installs printer driver, copies the files needed for driver, service and applications
- novaPDF11COM(x64).msi, novaPDF11COM(x86).msi - installs NovaPdfOptions110 COM
- novaPDF11SDK(x64).msi, novaPDF11SDK(x86).msi - installs configuration files (profiles database,
license,...) and adds a printer
- novaPDF11Tools.msi - installs Profile Manager tool

SDK\Lib
- novapi11.dll - INovaPdfOptions11 binary file. There are two versions of the dll, one for i386 systems and one for x64 systems

SDK\Tools
- novaPDF Co-Branding tool - customization tool for installers, see Customize your setup

In "C:\Users\Public\Documents\novaPDF 11\SDK" the installer will create the following folder structure:

Include
- definition files for INovaPdfOptions interface
- definitions for Windows messages and Profile option strings

Samples
Contains several samples of how to use INovaPdfOptions:
- Access PDF Reports - make a report on an Access database and convert it to PDF
- ASP.NET Hello World - an ASP.NET application that prints using the Printer object
- C++ Hello World - a console application that prints one page to the novaPDF SDK 11
- C++ Hello World (network) - the same as Hello World sample, but it can be run from any computer in the network, thought the novaPDF SDK 11 is installed on one single computer
- C++ MFC Scribble - the standard MFC Scribble sample extended with generate PDF files
- C++ MFC Converter - a MFC dialogs sample that converts an existing file to PDF using different profiles on novaPDF SDK 11
- C++ Override Options - a console application that uses override options instead of option profiles and prints one page to the novaPDF SDK 11
- C# Hello World CSharp - a simple Windows console application that prints one page to the novaPDF SDK 11.
- C# CSharp Converter - converts an existing file to PDF using different profiles on novaPDF SDK 11
- C# Word OLE CSharp - converts a document created with Microsoft Word to PDF using Word automation
- C# Override Options - a console application that uses override options instead of option profiles and prints one page to the novaPDF SDK 11
- Delphi Hello World Delphi - a Delphi application that prints using the Printer object
- Delphi VCL Converter - a Delphi application that converts an existing file to PDF using different profiles on novaPDF SDK 11
- Delphi Word OLE Delphi - converts a document created with Microsoft Word to PDF using Word automation
- Delphi Override Options - a console application that uses override options instead of option profiles and prints one page to the novaPDF SDK 11
- Java Hello World Java - an Java application that prints using the Printer object
- Java Word OLE (Java) - converts a document created with Microsoft Word to PDF using Word automation
- VBNVet Hello World VBNVet - a VBNVet console application that prints one page to the novaPDF SDK 11.
- VBNVet VBNVet Converter - a VBNVet application that converts an existing file to PDF using different profiles on novaPDF SDK 11
- VBNVet Word OLE VBNVet - converts a document created with Microsoft Word to PDF using Word automation
- VBNVet Override Options - a console application that uses override options instead of option profiles and prints one page to the novaPDF SDK 11
-Wix bundle - setup sample; compresses the msi files in an setup executable

Bin
- sample executables for DotNet, Win32 and Win64

1.2.4 Network use

novaPDF SDK 11 network auto-install
novaPDF SDK 11 can be installed on one computer and can be used by any computer in the network, without having to install it on each computer. This is to ease the work of network administrators both at installation time and future upgrades.

novaPDF SDK 11 supports Point and Print technology. This means that you can install the printer on one computer on the network, share it, and you can connect to it from any other computer. The system copies the necessary files for the driver, without any user interaction. On the server there are installed both i386 and x64 drivers and you can connect from the network with any i386 or x64 computers.

How to use novaPDF SDK 11 in a network
If you have a large network you can install novaPDF SDK 11 and your application which integrates novaPDF 11 SDK Developer on a single computer and use it from any computer in the network. All you need to do in your software is to initialize the INovaPdfOptions interface with the correct printer name, including the name of the computer on which it is installed (like "\server\novaPDF SDK 11").
When the application initiates the first print job to the printer server, the system copies the necessary printer driver files without any user interaction and the print job is completed on the printer server.

You can configure private or public profiles on the printer server. Public profiles will be visible on all client computers and all users. Private profiles are visible only for the user that created them. See Private and public profiles for more details.

The COM has to be installed and registered on every computer that uses it., see here How to register COM.

An alternative solution is to use the reg-free registration-free COM technology that enables your application to use the COM without registering it. You just have to import the COM manifest in your application and copy the COM dll in the same folder with your application executable. (you need the novapi11.dll and novasv11.dll).
1.2.5 **Multiple printers**

There can be added several printers, each of them using a different active profile. In this way, the users will decide to which printer to print instead of using the same printer and changing its active profile.

The best way to add multiple printers is at installation time. When running the Co-Branding Tool define all printers you wish to install. The msi that will be generated will add all these printer at installation.

NovaPDF printers can be added also manually from the Printer Manager tool. On the first page there can be added/deleted printers and on the second page there can be set which public profiles to be used with each printer. Take care that administrative privileges are required for adding / removing printers.

Printer Manager tool can also be run silently from command line. Take care to run it with administrative privileges.

To add a printer call the PrinterManager.exe tool, from the printer driver installation folder C: \Program Files\Softland\novaPDF 11\Driver. Call it for each printer with next command line parameters:

```
PrintManager.exe /silent /oem=<your OEM ID> /port=8501 /add /printer=<printer name> /printerport=<port name>
```

It is better to add each printer on a different port so they can work independently. When multiple printers work on the same port, the documents are processed sequentially.

To assign a default active profile for a printer call the Printer Manager tool with next command line parameters:

```
PrintManager.exe /silent /oem=<your OEM ID> /port=8501 /printer=<printer name> /set /profile=<profile id>
```

Or you could perform both steps in one call, add a printer and set the active profile

```
PrintManager.exe /silent /oem=<your OEM ID> /port=8501 /add /printer=<printer name> /printerport=<port name> /profile=<profile id>
```

You may see the profile id when you start the Profile Manager tool, General options page.

To remove a printer, call Printer Manager with these parameters:

```
PrintManager.exe /silent /oem=<your OEM ID> /port=8501 /remove /printer=<printer name>
```

Other optional command line parameters for PrinterManager.exe are:

```
/allowprivates=true (or false) - specify if the printer allows or not private profile to be created
```
/showselect=true (or false) - specify if Select Profile dialog should be shown when printing
/allowhide=true (or false) - specify if users are allowed to hide Select Profile dialog

1.3 Integration

1.3.1 How to integrate

You have to follow these steps for integrating novaPDF 11 SDK Developer in your application:

1. Install novaPDF 11 SDK Developer

   When installing novaPDF 11 SDK Developer, a "novaPDF SDK 11" printer is added in the Printers list in Control Panel.

2. Take a sample and test it

   See What sample to choose topic for directions how to choose the best sample for your situation.

3. Copy relevant code from the sample in your application

   Be sure you include all next steps from samples:
   
   - customize novaPDF SDK 11 settings using INovaPdfOptions11 COM interface (for instance set the output file name and folder, document info,...). See Profile options topic for a list of all option constants. There are global definition files for all supported programming languages.
   
   - start a print job and write to the printer device context (using functions like CreateDC, StartDoc, StartPage, TextOut,...). Or open a file and print it with other methods, like calling ShellExecute().

4. Test how your application prints to novaPDF SDK 11

   When you print to novaPDF SDK 11, the generated PDF files have an unlicensed notice on the bottom of the PDF pages. To remove this text please read the How to make the release build topic.

1.3.2 How to make the release build

After you succeeded to integrate novaPDF 11 SDK Developer in your application (see How to integrate topic) you have to follow next steps:

1. Purchase a novaPDF 11 SDK Developer license

   If you want to remove the novaPDF notice from the generated PDF files, you have to purchase a license. There are two types of application licenses:

   1. Software application license - this type of license allows you to to develop, market and distribute ONE program or ONE software product that integrates the novaPDF 11 SDK Developer to an unlimited number of end users without any additional fees.

   2. Component application license - This type of license allows you to to develop, market and distribute ONE component, ONE wrapper, ONE library or ONE module that integrates the novaPDF 11 SDK Developer to an unlimited number of end users without any additional fees.
After purchase, you will receive an email with the next information:

- license key
- licence file
- customization file

Each licensed user will received an unique ID that will guarantee your novaPDF 11 SDK Developer customized installers will install separately from the other users’.

2. Customize redistributable novaPDF installers with your license
With novaPDF 11 SDK Developer there is installed a tool called "novaPDF Co-Branding". You should run this tool to generate the msi files customized for your license. You can start this tool from the novaPDF 11 SDK Developer installation menu. See Customize your setup for more details. You will need the resulting customized setup files in step 3 below.

3. Install customized printer installation files
For testing purposes install novaPDF 11 SDK Developer running the resulting customized setup files (MSI) in the following order:

On Windows 32 bit computers:
- novaPDF11PrinterDriver(x86).msi - installs both 32 and 64 bit versions of the printer driver
- novaPDF11COM(x86).msi - installs the 32 bit version of the COM
- novaPDF11SDK(x86).msi - installs the custom, licensed printer
- novaPDF11Tools.msi - installs Profile Manager tool

On Windows x64 computers:
- novaPDF11PrinterDriver(x64).msi - installs both 32 and 64 bit versions of the printer driver
- novaPDF11COM(x64).msi - installs the 64 bit version of the COM
- novaPDF11COM(x86).msi - installs the 32 bit versions of the COM (only install this if your application is 32 bit)
- novaPDF11SDK(x64).msi - installs the custom, licensed printer
- novaPDF11Tools.msi - installs Profile Manager tool

You can install the msi silently using the following command line:
msiexec /i <msi file name> /qn

You can install the package bundle silently using the following command line:
<setup name> /q

To create the final release build you need to create an installation package bundle and run it. See the Build an installation package bundle topic.

4. Print without unlicensed notice
Your printer should be licensed now so the generated PDF files should not have the unlicensed footer notice.
If the footer notice is still showing up, you have to verify that you followed the steps from How to integrate correctly.
1.3.3 **Customize your setup**

Each developer that integrates a novaPDF 11 SDK Developer license will receive a unique OEM ID so its novaPDF installation does not interfere with other developers installation. In order to have an unique, licensed printer, you have to customize the setup that you will distribute with your application with the "novaPDF Co-Branding" tool. Run the tool and follow the wizard steps to generate the msi.

1.3.4 **Co-Branding Tool**

1.3.4.1 **Folders**

**Branding folder**
When starting the application, you will be asked for the branding folder. This is the folder where the new msi files will be generated. You can create a new folder or use an existing one. If you choose an existing folder, the files from the folder will be deleted and the original msi installers will be copied there and injected with the new properties.

**Settings file**
If you already have run the tool and you only wish to change some options, select the settings file from the previous run.

1.3.4.2 **Files**

There are several files that can be injected in the .msi:

**Customization (.ctm) and License (.lic) files.**

License file contains information about your license and Customization file is used for re-branding options. Contact novaPDF 11 SDK Developer support team for more information about re-branding.

When buying a the novaPDF 11 SDK Developer license will receive these two files. You should use them both to create a licensed printer.

If you do not have the novaPDF 11 SDK Developer license yet, select the "Use default Customizations (*.ctm) and License (*.lic) files" option to use some files that the novaPDF 11 SDK Developer provide for trial testing.

**Controls (.ctl) file**

Special customization file for the Profile Manager tool. Contact novaPDF 11 SDK Developer support team for more information about customizing Profile Manager tool.

**Forms (.nps) file**

This file contains the printer forms available for the printer. This is a text file and you can modify it to distribute the forms you wish. User defined forms should have a format like this: 0;0;281;36x48 inches;36x48 inches;914400;1219200;0;0;0;0; increase the form number on third position and specify form width and height in thousands of
millimeters)

**Resolutions (.npr) file**

This file contains the resolutions available for the printer. This is a text file and you can modify it to distribute the resolutions you wish. Resolutions should be values between 50 and 2400 and you can specify maximum 100 resolutions.

**EULA (*.rtf)**

Default EULA file is empty. You can select the EULA document for your application.

**Help (.chm) file**

If you wish to install a different help file in .chm format.

**Use only default profiles and presets**

It will be installed an empty profiles database, containing only default options. The default profiles database will overwrite existing profiles database from previous installation.

**Use profiles and presets you defined**

If you wish to distribute some public profiles with your application, you can define the public profiles by running the Profile Manager tool. The profiles database will be included in your customized msi.

**Overwrite existing profiles and presets**

If this option is set, the new profiles database will be copied over existing profiles database from previous installation so the new defined profiles will be available on upgrades installations too. If this option is not set, when upgrading over a previous installation the profiles database is not copied so are kept previous profiles.

**Default Files**

All files from Default files folder will be included in the msi. Add there all files (image watermarks, certificates, overlay files) you wish to distribute.

### 1.3.4.3 Settings

**Your application name**

The application name will appear in Windows Start menu, in Control Panel \ Programs and on the About page of the installed printer and tools.

**Your company name**

The company name will appear on the About page of the installed printer and tools.

**Language code**
Language for novaPDF printer and installed tools.

**Add a start menu folder**
Enter the name of the Windows menu. You can also choose what links to be added in the menu (Printer Manager tool, Printer Monitor tool and Help file).

**Add Printer**
The installer can add one or multiple printers at installation. You can also choose to not add any printer at installation time. For each printer you can configure:
- printer name
- printer port name
- if the printer should be set default printer on the system
- associate a public profile with the printer
- printer defaults (default paper size, resolution, orientation, scale and others)
- printer permissions (permissions to change paper size, resolution, orientation, scale and others)

**Inject**
When you finish the configuration press the Inject button to create the msi files. There will be shown a dialog asking if you wish to save the configured options in a file so you can reuse it when running again the Co-Branding tool.

### 1.3.4.3.1 Add/Modify Printer

#### 1.3.4.3.1.1 General
For each printer you can configure:

**Printer name**
The name of the printer that you wish to install with your application.

**Port name**
The name of the port for the printer. Choose a name specific to your application so it will be unique. If you add multiple printers, it is better to put them on different ports, otherwise the documents will be processed sequentially.

**Share printer**
You can share the printer in the network and choose the share printer name.

**Make printer default**
You can also choose if the printer should be set as the default printer on the system.
1.3.4.3.1.2 Profile

Allow users to use private profiles

If users can create private profiles with the Profile Manager tool and use them when converting documents.

Show Select Profile dialog before printing

Force the show of Select Profile dialog before printing a document for all users. You can allow users to disable the dialog or not.

GUID of the active profile

The GUID of the public profile that you wish to set as default for the installed printer. You can copy the GUID from the Profile Manager, when it is opened in the Administrative mode (from the Printer Manager tool).
This parameter is not mandatory.

1.3.4.3.1.3 Page defaults

The default page size for the printer can be a predefined paper or a custom paper size.

Use existing page size

Choose from one of the predefined paper sizes.
The page sizes can be changed in the Forms (.nps) file

Define new page size

You can specify a new page size to be added when installed. Enter paper name, description, width and height. This new page size will be the default page size for the printer.

1.3.4.3.1.4 Other defaults

Enter printer specific default options:

Orientation

Default printer orientation can be Portrait or Landscape.

Resolution

Default resolution. Usually 300 or 600.

Scale

Default printer scale percent. Usually 100.

Copies
How many page copies to create. Usually 1.

**Collate**

If creating multiple pages, in what order to add the copies.

**Maximum copies**

How many copies should be maximum allowed.

### 1.3.4.3.1.5 Permissions

Users can be restricted to modify certain printer options:

- page size
- orientation
- resolution
- scale
- copies
- collate

### 1.3.4.4 Results

When pressing the inject button the new customized msi files will be created. The settings you filled in these pages can be saved in an ini file in the branding folder.

After running the tool, next msi files will be available in the branding folder:
- novaPDF11PrinterDriver(x64).msi or novaPDF11PrinterDriver(x86).msi - each of them installs both 32 and 64 bit versions of the printer driver
- novaPDF11Tools.msi - installs Profile Manager tool
- novaPDF11COM(x64).msi or novaPDF11COM(x86).msi - install 32 and 64 bit versions of the COM
- novaPDF11SDK(x64).msi or novaPDF11SDK(x86).msi - install the custom, licensed printer

**Note**

After the msi files are injected with your custom parameters, their digital signature becomes invalid so they need to be signed again. See more details in the topic Build an installation package bundle

### 1.3.5 Build an installation package bundle

If you already have an installation package for your application, you only need to add the novaPDF 11 SDK Developer msi files to your package.

If not, you can start with the installation package sample we included with novaPDF 11 SDK Developer. Follow the steps below to create the bundle:
1. Install WIX Toolset 3.9
This is a free and open source set of tools for building Windows installation packages. You will use this to create the bundle.

2. Sign the novaPDF 11 SDK Developer msi files
All msi files have to be signed with a valid digital signature. You can sign them by running the `signtool.exe` tool and providing your company digital signature file.
If you do not have a digital signature file you can upload the msi files on our site to be signed with Softland's signature. Log in to your account, go to your license key and press CODESIGN REQUEST button:
https://www.novapdf.com/myaccount.login.html

You will be requested to enter your novaPDF 11 SDK Developer license key, your email address and the path to the msi file.
You have to upload and sign these two msi files:
- novaPDF11SDK(x64).msi
- novaPDF11SDK(x86).msi

You will receive an email with the download link for the signed msi files.

3. Open the Installation package bundle sample and build the setup executable.
In the sample you have to change the defined variables with information about your application:
application name, company name, version, company site url.
**It is very important to change the Upgrade code to a new generated GUID, because each application should have its unique GUID so it installs separately on Windows.**

Also, take care to set the correct path of the novaPDF 11 SDK Developer msi files.

If you wish to sign the bundle exe, you have to use your company's signature. Use the signtool.exe and the insignia.exe tools to sign, like this:

```
"C:\Program Files (x86)\WIX Toolset v3.9\bin\insignia.exe" -ib "C:\Users\Public\Documents\novaPDF 11\OEM\Samples\OEMBundle\OEMBundle\bin\Release\OEMBundle.exe" -o "C:\Users\Public\Documents\novaPDF 11\OEM\Samples\OEMBundle\OEMBundle\bin\Release\tmp.exe"
"signtool.exe" sign /f "<your signature file>" /p "<your signature password>" /du "<your company site>" /t http://timestamp.verisign.com/scripts/timestamp.dll "C:\Users\Public\Documents\novaPDF 11\OEM\Samples\OEMBundle\OEMBundle\bin\Release\tmp.exe"
"C:\Program Files (x86)\WIX Toolset v3.9\bin\insignia.exe" -ab "C:\Users\Public\Documents\novaPDF 11\OEM\Samples\OEMBundle\OEMBundle\bin\Release\tmp.exe" "C:\Users\Public\Documents\novaPDF 11\OEM\Samples\OEMBundle\OEMBundle\bin\Release\OEMBundle.exe" -o "C:\Users\Public\Documents\novaPDF 11\OEM\Samples\OEMBundle\OEMBundle\bin\Release\OEMBundle.exe"
```
1.3.6 **Installation package bundle**

This is a sample on how to compress the msi installations packages in a bundle and generate an setup executable to install novaPDF 11 SDK Developer.

If you haven’t already done so, you will need to install the Wix Toolset to build this project.

The project contains two files:

1. **Variables.wxi** - contains variables for application name, company name, version,...; it's important to change the UpgradeCode to an unique new GUID for your application
2. **Bundle.wxs** - contains the msi files that need to be included and the execution order; .Net Framework 4 is required by novaPDF 11 SDK Developer and is set as prerequisite

**Variables.wxi**

```xml
<?xml version="1.0" encoding="utf-8"?>
<Include>

<?define BundleName="My SDK Application Bundle"?>

<?define AboutUrl='http://www.novapdf.com'?>
<?define SplashImage=" "?>
<?define IconFile=" "?>

<?define MyLicenseFileName="C:\ProgramData\Softland\novaPDF 11\PdfSdk11_Softland\PdfSdk11_SoftlandEulaExt.rtf" ?>

<?define MajorVersion="11"?>
<?define MinorVersion="0"?>
<?define BuildNumber="1"?>

<?define Manufacturer="<My SDK Company Name>" ?>

<?define ProductName="<My SDK Application Name>"?>

<?define FullProductName="$\{var.ProductName\} $\{var.MajorVersion\}" ?>

<?define DriverKit86="C:\Users\Public\Documents\novaPDF 11\SDK\branding\novaPDF11PrinterDriver(x86).msi"?>
<?define DriverKit64="C:\Users\Public\Documents\novaPDF 11\SDK\branding\novaPDF11PrinterDriver(x64).msi"?>

<?define COMPPath64="C:\Users\Public\Documents\novaPDF..."?>
```
<?define COMPathx86="C:\Users\Public\Documents\novaPDF 11\SDK\Branding\novaPDF11COM(x86).msi"?>
<?define OemKitx86="C:\Users\Public\Documents\novaPDF 11\SDK\Branding\novaPDF11SDK(x86).msi"?>
<?define OemKit64="C:\Users\Public\Documents\novaPDF 11\SDK\Branding\novaPDF11SDK(x64).msi"?>
<?define ToolsKitx="C:\Users\Public\Documents\novaPDF 11\SDK\Branding\novaPDF11Tools.msi"?>
</Include>

Bundle.wxs

<?xml version="1.0" encoding="UTF-8"?>
<wix xmlns="http://schemas.microsoft.com/wix/2006/wi"
     xmlns:util="http://schemas.microsoft.com/wix/UtilExtension"
     xmlns:bal="http://schemas.microsoft.com/wix/BalExtension"
     xmlns:swid="http://schemas.microsoft.com/wix/TagExtension"
     xmlns:dotNet="http://schemas.microsoft.com/wix/NetFxExtension">
</wix>

<?include "Variables.wxi"?>
</Bundle>

<swid:Tag Regid="regid.2008-09.org.wixtoolset" />

<!--change the UpgradeCode to an unique new GUID for your application-->  
<RelatedBundle Id="$(var.UpgradeCode)" Action="Upgrade" />

<!--application license-->  
<BootstrapperApplicationRef Id="WixStandardBootstrapperApplication.Rtflicense"
   bal:WixStandardBootstrapperApplication LicenseFile="$(var.MyLicenseFileName)"
   SuppressOptionsUI="yes"/>
</BootstrapperApplicationRef>

<Chain>
<!--prereqs-->  
<PackageGroupRef Id="NetFx40Web"/>

<!--COM-->  
CacheId="COMIdx86\$(var.MajorVersion).\$(var.MinorVersion).\$(var.BuildNumber)"
DisplayInternalUI="no"
EnableFeatureSelection="no" ForcePerMachine="yes"
Compressed="yes" SourceFile="\$(var.COMPathx86)" Visible="yes" Vital="yes">
  </MsiPackage>
</!---driver-->
<Msipackage Id="DriverPackage\$(var.MajorVersion).\$(var.MinorVersion).\$(var.BuildNumber)"
CacheId="COMIdx86\$(var.MajorVersion).\$(var.MinorVersion).\$(var.BuildNumber)"
DisplayInternalUI="no"
EnableFeatureSelection="yes" ForcePerMachine="yes"
Compressed="yes" SourceFile="\$(var.DriverKit86)" Visible="yes"
Vital="yes" InstallCondition="NOT VersionNT64">
  </Msipackage>
</Msipackage>
<Msipackage Id="DriverPackage\$(var.MajorVersion).\$(var.MinorVersion).\$(var.BuildNumber)"
CacheId="COMIdx86\$(var.MajorVersion).\$(var.MinorVersion).\$(var.BuildNumber)"
DisplayInternalUI="no"
EnableFeatureSelection="yes" ForcePerMachine="yes"
Compressed="yes" SourceFile="\$(var.DriverKit64)" Visible="yes"
Vital="yes" InstallCondition="VersionNT64">
  </Msipackage>
</Msipackage>
<Msipackage Id="NovaPDFTools\$(var.MajorVersion).\$(var.MinorVersion).\$(var.BuildNumber)"
CacheId="COMIdx86\$(var.MajorVersion).\$(var.MinorVersion).\$(var.BuildNumber)"
DisplayInternalUI="no"
EnableFeatureSelection="no" ForcePerMachine="yes"
SuppressSignatureVerification="yes"
Compressed="yes" SourceFile="\$(var.ToolsKit)" Visible="yes" Vital="yes">
  </MsiPackage>
</!---oem product-->
1.4 novaPDF COM

1.4.1 How to register COM

novaPDF SDK includes a COM interface, INovaPdfOptions. The COM binary file is located in the Lib sub folder. The COM is first registered when installed with novaPDF SDK. If you want to register/unregister novaPDF COM manually, use the following commands from the command line:

Register
regsvr32.exe "C:\Program Files\Softland\novaPDF 11\SDK\Lib\i386\novapi11.dll"

or, for x64 systems:
regsvr32.exe "C:\Program Files\Softland\novaPDF 11\SDK\Lib\x64\novapi11.dll"

Unregister
regsvr32.exe /u "C:\Program Files\Softland\novaPDF 11\SDK\Lib\i386\novapi11.dll"
or, for x64 systems:
regsvr32.exe /u "C:\Program Files\Softland\novaPDF 11\SDK\Lib\x64\novapi11.dll"

msi
In the "C:\Program Files\Softland\novaPDF 11\SDK\Installer" folder there are two msi that install the x86 or x64 version of the COM and register it. These only install the COM and they are meant to be used on client computers if you install novaPDF printer on a server and run your application from different computers:
novaPDF11COM(x86).msi
anovaPDF11COM(x64).msi

1.4.2 How to use the COM
To use novaPDF COM in your application you need to follow next steps:

1. Create an instance of INovaPdfOptions interface
Create the object when the application starts and use it while printing to novaPDF SDK 11 printer

2. Call the Initialize method
Initialize method has one parameter, the name of the printer (for example "novaPDF SDK 11", or when on the network "\server name\novaPDF SDK 11")

3. Set novaPDF SDK 11 options by calling SetOptionString or SetOptionLong (...) methods.
You can also manage profiles with AddProfile, CopyProfile, DeleteProfile, GetFirstProfile, GetNextProfile, GetActiveProfile, SetActiveProfile methods. A good sample for how to use this methods is the MFC Converter sample. Also, all "Hello World" samples have a "nova" unit where there are samples on how to set all options.
This step is optional. If you use the default options or if you already configured the desired options, you can skip it.

4. Register your application to receive Windows messages
novaPDF printer sends messages (StartDoc, StartPage, EndPage, EndDoc, FileSent, Print Error) while printing a document. You can register to receive Windows messages using the RegisterEventWindow method. You also need to implement message handlers for the registered messages. See MFC Scribble or MFC Converter samples.
This step is also optional. If you do not need to implement this event handlers you can skip it.

5. Start a print job. You can do it as follows:
   • use Win32 API functions: OpenPrinter, DocumentProperties, CreateDC, StartDoc, StartPage,...
     See the Hello World sample.
   • print a file using the ShellExecute function. For a sample see MFC Converter.
   • use MFC document/view architecture. For more information look at the MFC Scribble sample.

6. Release the INovaPdfOptions instance
The object should be released only when all print jobs are finished.
1.4.3 How to set printer options

You can use INovaPdfOptions interface to read or set novaPDF SDK 11 options.

INovaPDFOptions provides the following methods for this:
GetOptionString
SetOptionString
GetOptionLong
SetOptionLong
SetOptionBool
GetOptionBool

The options are saved in the current loaded profile. See Private and public profiles topic for more details about profiles.

You have to make these settings before starting the print job.

You can find the complete list of option constants that you can use in the GetOptionXXXX and SetOptionXXXX in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

Before setting the options the profile should be loaded with LoadProfile and when finish the profile should be saved with SaveProfile. Before printing a document, set the profile you wish to use with SetActiveProfile.

1.4.4 Private and public profiles

Public profiles are visible from all client computers and from all users. You should use public profiles if you want to configure printing options that should be used by several computers in your network. For instance, you can configure a folder where all client computers will save the PDF files.

Private profiles are visible only to the user that created them.

1.4.5 How to register for messages

novaPDF SDK 11 generates the following events when processing a print job:
- Document Started
- Page Started
- Page Ended
- Document Ended
- File Sent
- File Saved
- Print Error
- Email Sent
• Email Error

When the events are fired by the driver, these Windows messages are sent to a registered window handler. To receive a message you need to register your window handle with the RegisterEventWindow method. See the MFC Converter and MFC Scribble samples.

These events are fired when the print job is processed by the driver. When an applications sends a print job to the printer, the print job is added in the printer spooler queue. There might be other jobs waiting in the queue that have to be finished before your job starts. So there might be some delay between the moment the application is sending the job to the printer, the moment the job is actually processed and the time when the PDF file is saved. If you need for instance to open the PDF file afterwards, you need to register to the File Saved message and process the PDF file in this message handler.

1.4.6 How to use events

If you want to wait for a print job to be finished, you can register some Windows events and wait for them to be signaled by novaPDF SDK 11.

Before starting the print job, you have to inform novaPDF SDK 11 that you want to be wait for an event.

Call RegisterNovaEvent(LPCWSTR p_wsEventName) with one one of the next strings:
   "NOVAPDF_EVENT_START_DOC"
   "NOVAPDF_EVENT_END_DOC"
   "NOVAPDF_EVENT_FILE_SAVED"

After you send the job to the printer, call WaitForNovaEvent(ULONG nMilliseconds, BOOL* p_bTimeout). This function will return when the event was signaled or when the time was elapsed. If the time was elapsed p_bTimeout is TRUE and if the event was signaled, p_bTimeout is FALSE.

If you just want to be sure that the print job was started, the profile was read, and you want to proceed modifying the profile for the next job, you should wait for the NOVAPDF_EVENT_START_DOC event. If you are interested where the PDF file is ready so you can do further actions with it, you should wait for the NOVAPDF_EVENT_FILE_SAVED event.

1.4.7 Multithreading applications

If you wish to use novaPDF in multi-threading or multi-process applications, there are some restrictions.

novaPDF COM is not thread safe, you should implement your own thread synchronization when using the COM.

When you use third party applications to print, they have to be licensed separately by using the LicenseApplication call. Because this call does not license this application for any print calls, but just for the next print job that will be executed by the printer, the calls of LicenseApplication and the execution of a print job must go in pair. If you call several times LicenseApplication in your code, but the printer processes some of the jobs later, they might not be licensed. So to assure these pair executions use next two calls when printing:
Before starting the print job call:
RegisterNovaEvent("NOVAPDF_EVENT_START_DOC")

After sending the print job, wait for the printer to start processing it and read the licensing for it by calling:
WaitForNovaEvent(-1, &bTimeout)

1.4.8 Use temporary printers

If you do not wish a novaPDF printer visible in the Printers list, you could use this approach:

When customizing your installer, do not add a printer (see Customize your setup)
What this means is that only the novaPDF printer driver and the COM are installed, but no printer is added in the Printers list.

In your application code, when you want to use novaPDF to convert to PDF:
- first call AddNovaPrinter to add a temporary printer
- set the printer options using the SetOptionXXX methods
- print a document, or several documents to generate PDF files
- delete the temporary printer with DeleteNovaPrinter

You may call the AddNovaPrinter and DeleteNovaPrinter several times, the only restriction is to call it in pair so you do not leave unused printers in the printers list.

When deleting a printer there might happen that the jobs are not all processed yet and there are still jobs in the printer queue. If you wish those PDF file to be generated, then is better to wait for the PDF to be finished before deleting the printer. It is recommended to use the event methods we provide (How to use events).

Network printing

If you wish to install novaPDF printer on a server and share it in the network, then a printer has to be added at installation time on the server, or the AddNovaPrinter and DeleteNovaPrinter must be called from an application running on server. This solution works on client computers by adding a printer connection to the printer on server instead of adding a local printer. So in this situation these are the changes:
- the installer must be run on the server normally, installing a printer there
- the application code is the same, only the printer name must contain the server path, like this: \ \<server name>\<printer name>, where the <printer name> printer must exist on server

Restrictions

On some operating systems you need administrative rights to add/delete a printer. Below you can see a list of operating systems supported by novaPDF, and what type of user accounts are working:

<table>
<thead>
<tr>
<th>System</th>
<th>Administrator</th>
<th>Power user</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows XP</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>
### 1.4.9 Working with Layout

In the Profile Manager tool there is a page called Layout where different objects can be placed and arranged with anchors on the page. There can be added image watermarks, text watermarks, overlay pdf document, signature. Also the printed page content itself can be arranged how to fit on the PDF page.

For each object there can be specified the anchors relative to the page: left, right, top, bottom, center vertically or center horizontally. The values can be absolute or in percentage relative to the page width and height.

When adding such an object from source code, you have to specify all these settings that you can also find on the Layout page (anchors flags and values, size, rotation, keep aspect ratio flag). If you wish to add one of these objects in the profile, configure first how it should appear on page using the Profile Manager tool, then copy generated settings in the source code.

### 1.4.10 Working with Actions

Action is a notion introduced in version 9 of novaPDF. An action is a task that novaPDF printer driver will execute before or after generating the pdf file. An action can be:
- copy or delete a file
- send an email or upload to ftp/sftp
- open the pdf in the default viewer
- run another application

Actions are saved in a list and they can be arranged in any order. By default, each profile contains two action:
- the "Save PDF" action is the one that generated the pdf file; this action cannot be removed from profile and it is added in the actions list as a referring item, so other actions can be added before or after this action depending at which moment the action should be run
- an "Open PDF" action that will open the PDF after it is generated; this is the default behavior but this action can be disabled if you do not wish to open the PDF

You can:
- add new actions in the list with an AddAction call
- change action properties with SetActionOptionString, SetActionOptionLong,.... for specific action settings
- change the order in which the actions are executed using the SetActionOptionLong(pwsActionID,
NOVAPDF_ACTION_INDEX, index);
- enable or disable an action without removing it from the list with EnableAction and DisableAction
- enable or disable all actions of a specific type (for instance disable all open actions) with
  DisableActionType and EnableActionType
- remove actions from the list with DeleteAction

If you wish to not open the PDF after it is generated simply disable all open actions like this:

pNova.DisableActionType(NOVA_ACTION_OPEN)

1.4.11 Reference
1.4.11.1 Profile option strings

* novaPDF SDK 11 * settings are saved in a database on the server. The profiles are available for all printers.

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_COMPRESS_ENABLE</td>
<td>9</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_TEXT_GRAPHICS</td>
<td>63</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_HIGH_COLOR</td>
<td>54</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_MONOCHROM</td>
<td>60</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_INDEXED</td>
<td>57</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_TEXT_GRAPHICS_TYPE</td>
<td>64</td>
<td>long</td>
<td>0 - zip compression</td>
<td>0</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_TEXT_GRAPHICS_LEVEL</td>
<td>65</td>
<td>long</td>
<td>1-9</td>
<td>5</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_HIGH_COLOR_TYPE</td>
<td>55</td>
<td>long</td>
<td>0 - zip compression, 1 - JPEG compression</td>
<td>1</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_HIGH_COLOR_LEVEL</td>
<td>56</td>
<td>long</td>
<td>1-9</td>
<td>7</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>Constant name</td>
<td>Value</td>
<td>Type</td>
<td>Possible values</td>
<td>Default</td>
<td>Function</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>-------</td>
<td>--------</td>
<td>---------------------------------------------------------------------------------</td>
<td>---------</td>
<td>------------------</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_INDEXED_TYPE</td>
<td>58</td>
<td>long</td>
<td>0 - zip compression</td>
<td>0</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_INDEXED_LEVEL</td>
<td>59</td>
<td>long</td>
<td>1-9</td>
<td>5</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_MONOCHROM_TYPE</td>
<td>61</td>
<td>long</td>
<td>0 - zip compression</td>
<td>0</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_COMPRESS_MONOCHROM_LEVEL</td>
<td>62</td>
<td>long</td>
<td>1-9</td>
<td>5</td>
<td>SetOptionLong</td>
</tr>
</tbody>
</table>

**GRAPHICS**

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_GR_DOWNSML_ENABLE</td>
<td>11</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_DOWNSAMPLE_HIGH_COLOR</td>
<td>84</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_DOWNSAMPLE_RES_HIGH_COLOR</td>
<td>86</td>
<td>long</td>
<td>72 - 2400</td>
<td>96</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_DOWNSAMPLE_TYPE_HIGH_COLOR</td>
<td>85</td>
<td>long</td>
<td>0 - BOX Filter 1 - BILINEAR Filter 2 - BSPLINE Filter 3 - BICUBIC Filter 4 - CATMULLROM Filter 5 - LANCZOS3 Filter</td>
<td>3</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_DOWNSAMPLE_INDEXED</td>
<td>87</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_DOWNSAMPLE_RES_INDEXED</td>
<td>89</td>
<td>long</td>
<td>72 - 2400</td>
<td>96</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_DOWNSAMPLE_TYPE_INDEXED</td>
<td>88</td>
<td>long</td>
<td>0 - BOX Filter 1 - BILINEAR Filter 2 - BSPLINE Filter 3 - BICUBIC Filter 4 - CATMULLROM Filter 5 - LANCZOS3 Filter</td>
<td>3</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_DOWNSAMPLE_MONOCHROM</td>
<td>90</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_DOWNSAMPLE_RES_MONOCHROM</td>
<td>92</td>
<td>long</td>
<td>72 - 2400</td>
<td>96</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_DOWNSAMPLE_TYPE_MONOCHROM</td>
<td>91</td>
<td>long</td>
<td>0 - BOX Filter 1 - BILINEAR Filter</td>
<td>3</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>Variable</td>
<td>Value</td>
<td>Type</td>
<td>Default Value</td>
<td>Set Option Function</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------</td>
<td>---------</td>
<td>---------------</td>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_GR_CONVERT_ENABLE</td>
<td>10</td>
<td>bool</td>
<td>false</td>
<td>SetOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_CONVERT_HIGH_COLOR</td>
<td>71</td>
<td>bool</td>
<td>false</td>
<td>SetOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_CONVERT_INDEXED</td>
<td>76</td>
<td>bool</td>
<td>false</td>
<td>SetOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_CONVERT_TYPE_HIGH_COLOR</td>
<td>73</td>
<td>long</td>
<td>0 - Grayscale 1 - Monochrome</td>
<td>SetOptionLong</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_DITHER_HIGH_COLOR</td>
<td>72</td>
<td>bool</td>
<td>true</td>
<td>SetOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_DITHER_TYPE_HIGH_COLOR</td>
<td>75</td>
<td>long</td>
<td>0 - FS Dither 1 - BAYER4 Dither 2 - BAYER8 Dither 3 - ORDER6 Dither 4 - ORDER8 Dither 5 - ORDER16 Dither</td>
<td>SetOptionLong</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_CONVERT_TYPE_INDEXED</td>
<td>78</td>
<td>long</td>
<td>0 - Grayscale 1 - Monochrome</td>
<td>SetOptionLong</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_DITHER_INDEXED</td>
<td>77</td>
<td>bool</td>
<td>true</td>
<td>SetOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_DITHER_TYPE_INDEXED</td>
<td>80</td>
<td>long</td>
<td>0 - FS Dither 1 - BAYER4 Dither 2 - BAYER8 Dither 3 - ORDER6 Dither 4 - ORDER8 Dither 5 - ORDER16 Dither</td>
<td>SetOptionLong</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_CONVERT_TEXTANDGR</td>
<td>81</td>
<td>bool</td>
<td>false</td>
<td>SetOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_CONVERT_TYPE_TEXTANDGR</td>
<td>82</td>
<td>long</td>
<td>0 - Grayscale 1 - Monochrome</td>
<td>SetOptionLong</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_CONVERT_TRESH_TEXANDGR</td>
<td>83</td>
<td>long</td>
<td>0 - 255</td>
<td>128</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_CONVERT_TRESH_HIGHTЕНEDCOLOR</td>
<td>74</td>
<td>long</td>
<td>0 - 255</td>
<td>128</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_IMAGE_CONVERT_TRESH_INDEXED</td>
<td>79</td>
<td>long</td>
<td>0 - 255</td>
<td>128</td>
<td>SetOptionLong</td>
</tr>
</tbody>
</table>

**FONTS**


<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_FONTS_EMBED_ALL_USED_FONTS</td>
<td>160</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_FONTS_EMBED_SUBSETS</td>
<td>159</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_FONTS_FORCE_EMBED_PROTECTED</td>
<td>161</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
</tbody>
</table>

**DOCUMENT INFO AND VIEWER OPTIONS**

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_INFO_ENABLE</td>
<td>7</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SHOW_DOCINFO_DLG</td>
<td>16</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_DOCINFO_AUTHOR</td>
<td>69</td>
<td>string</td>
<td>[U]</td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_DOCINFO_CREATOR</td>
<td>67</td>
<td>string</td>
<td>[N]</td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_DOCINFO_KEYWORDS</td>
<td>70</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_DOCINFO_SUBJECT</td>
<td>68</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_DOCINFO_TITLE</td>
<td>66</td>
<td>string</td>
<td>[N]</td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_INFO_VIEWER_ENABLE</td>
<td>8</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_PAGE_LAYOUT</td>
<td>34</td>
<td>long</td>
<td>0 - Viewer default 1 - single page 2 - column 3 - two columns, left 4 - two columns, right 5 - two pages, left 6 - two pages, right</td>
<td>0</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_PAGE_MODE</td>
<td>38</td>
<td>long</td>
<td>0 - Viewer default 1 - None 2 - show bookmarks panel 3 - show pages panel 4 - Full screen 5 - show layers</td>
<td>0</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
<td>Value</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_START_PAGE</td>
<td>long</td>
<td>33</td>
<td>true SetOptionLong; display start page panel;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_ZOOM</td>
<td>long</td>
<td>31</td>
<td>0 - Default viewer settings; 1 - Default 2 - Fit Width; 3 - Fit Height; 4 - Fit Page; 5 - Percent; 6 - Fit Visible</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_USE_START_PAGE</td>
<td>bool</td>
<td>32</td>
<td>false SetOptionBool;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_PAGE_SCALING</td>
<td>long</td>
<td>40</td>
<td>0 - Application Default 1 - None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_HIDE_TOOLBARS</td>
<td>bool</td>
<td>25</td>
<td>false SetOptionBool;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_SHOW_DOCTITLE</td>
<td>bool</td>
<td>26</td>
<td>false SetOptionBool;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_TRANSITION_TYPE</td>
<td>long</td>
<td>27</td>
<td>0 - Viewer default 1 - Replace 2 - Split 3 - Blinds 4 - Box 5 - Wipe 6 - Dissolve 7 - Glitter 8 - Fly in 9 - Fly out 10 - Push 11 - Cover 12 - Uncover 13 - Fade</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_TRANSITION_DIRECTION</td>
<td>long</td>
<td>28</td>
<td>0 - Horizontal 1 - vertical</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_TRANSITION_DURATION</td>
<td>long</td>
<td>29</td>
<td>0 SetOptionLong;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_PAGE_DURATION</td>
<td>long</td>
<td>30</td>
<td>0 SetOptionLong;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_HIDE_MENUBAR</td>
<td>bool</td>
<td>35</td>
<td>false SetOptionBool;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_RESIZE_WIND</td>
<td>bool</td>
<td>36</td>
<td>false SetOptionBool;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Constant name</td>
<td>Value</td>
<td>Type</td>
<td>Possible values</td>
<td>Default</td>
<td>Function</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------</td>
<td>--------</td>
<td>--------------------------------------------------------------------------------</td>
<td>---------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>NOVAPDF_SEC_ENABLE</td>
<td>13</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SHOW_SECURITY_DLG</td>
<td>15</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SECURITY_FLAG_PRINTDOC</td>
<td>112</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SECURITY_FLAG_MODIFYDOC</td>
<td>113</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SECURITY_FLAG_EXTRACTTEXT</td>
<td>114</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SECURITY_FLAG_MODIFYANNOTATION</td>
<td>115</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SECURITY_FLAG_ADVANCEDFILL</td>
<td>119</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SECURITY_FLAG_ADVANCEDEXTRACT</td>
<td>118</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SECURITY_FLAG_ASSEMBLEDOC</td>
<td>117</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SECURITY_FLAG_PRINTHIGHRES</td>
<td>116</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SECURITY_USER_PASSWORD</td>
<td>110</td>
<td>string (encrypted)</td>
<td></td>
<td>SetOptionEncryptedString</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_SECURITY_OWNER_PASSWORD</td>
<td>111</td>
<td>string (encrypted)</td>
<td></td>
<td>SetOptionEncryptedString</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_SECURITY_LEVEL</td>
<td>109</td>
<td>int</td>
<td>1 - 40 bits RC4</td>
<td>4</td>
<td>SetOptionLong</td>
</tr>
</tbody>
</table>

**OW**

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_VIEWOP_HIDE_USER</td>
<td>37</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_VIEWOP_FULLSCREEN</td>
<td>39</td>
<td>long</td>
<td>0 - Viewer default&lt;br&gt;1 - None&lt;br&gt;2 - Bookmarks&lt;br&gt;3 - Pages&lt;br&gt;4 - Layers</td>
<td>0</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>Constant name</td>
<td>Value</td>
<td>Type</td>
<td>Possible values</td>
<td>Default</td>
<td>Function</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-------</td>
<td>-------</td>
<td>-----------------------------------</td>
<td>---------</td>
<td>--------------------</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_CERTIFICATE_TYPE</td>
<td>131</td>
<td>long</td>
<td>1 - system certificate</td>
<td></td>
<td>SetOptionLong</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 - certificate file</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SYSCERT_ISSUED_TO</td>
<td>125</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SYSCERT_ISSUED_BY</td>
<td>126</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SYSCERT_FRIENDLY_NAME</td>
<td>127</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SYSCERT_SUBJECT</td>
<td>128</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SYSCERT_SUBJECT_NAME</td>
<td>129</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SYSCERT_EXP_DATE</td>
<td>130</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_FILECERT_LOCATION</td>
<td>120</td>
<td>long</td>
<td>1 - Local</td>
<td>1</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 - Server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_SIGN_FILECERT_FILE</td>
<td>121</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_FILECERT_FILE_PASSWORD</td>
<td>122</td>
<td>string(encrypted)</td>
<td></td>
<td>SetOptionEncryptedString</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_SIGN_FILECERT_USER_NAME</td>
<td>123</td>
<td>string</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_SIGN_FILECERT_USER_PASSWORD</td>
<td>124</td>
<td>string(encrypted)</td>
<td></td>
<td>SetOptionEncryptedString</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SHOW_GRAPHIC_NAME</td>
<td>132</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SHOW_GRAPHIC_IMAGE</td>
<td>133</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_IMAGE_LOCATION</td>
<td>148</td>
<td>long</td>
<td>1 - Local</td>
<td>1</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 - Server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_SIGN_IMAGE_FILE</td>
<td>149</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_IMAGE_USER_NAME</td>
<td>150</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_IMAGE_USER_PASSWORD</td>
<td>151</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionEncryptedString</td>
</tr>
<tr>
<td>Constant name</td>
<td>Value</td>
<td>Type</td>
<td>Possible values</td>
<td>Default</td>
<td>Function</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>--------</td>
<td>-----------</td>
<td>--------------------------</td>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SHOW_KEEPASPECT_RATIO</td>
<td>134</td>
<td>bool</td>
<td>true</td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_IMAGE_OPACITY</td>
<td>152</td>
<td>long</td>
<td>100</td>
<td>100</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_NAME_OPACITY</td>
<td>147</td>
<td>long</td>
<td>100</td>
<td>100</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SHOW_CERTIFICATE_NAME</td>
<td>135</td>
<td>bool</td>
<td>false</td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SHOW_SIGN_DATE</td>
<td>136</td>
<td>bool</td>
<td>false</td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SHOW_SIGN_LOCATION</td>
<td>139</td>
<td>bool</td>
<td>false</td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SHOW_SIGN_REASON</td>
<td>138</td>
<td>bool</td>
<td>false</td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SHOW_LABELS</td>
<td>141</td>
<td>bool</td>
<td>false</td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SHOW_DETAILS</td>
<td>140</td>
<td>bool</td>
<td>true</td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_SHOW_CONTACT_INFO</td>
<td>137</td>
<td>bool</td>
<td>false</td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_FONT_NAME</td>
<td>142</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_FONT_TYPE</td>
<td>143</td>
<td>long</td>
<td>0 - TrueType 1 - Type1 2 - OpenType (TrueType) 3 - OpenType (Tpe1)</td>
<td>0</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_CUSTOM_SIZE</td>
<td>145</td>
<td>bool</td>
<td>false</td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_FONT_SIZE</td>
<td>144</td>
<td>long</td>
<td>8</td>
<td>8</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_NAME_COLOR</td>
<td>146</td>
<td>long</td>
<td>RGB color</td>
<td>RGB color</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_INFO_REASON</td>
<td>154</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_INFO_LOCATION</td>
<td>155</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_INFO_CONTACT</td>
<td>153</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_VIEW_VIEW</td>
<td>156</td>
<td>bool</td>
<td>true</td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_VIEW_PRINT</td>
<td>157</td>
<td>bool</td>
<td>true</td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_SIGN_VIEW_EXPORT</td>
<td>158</td>
<td>bool</td>
<td>true</td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
</tbody>
</table>
### NOVAPDF_URL ANALYZE

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_URL_ANALYZE</td>
<td>5</td>
<td>bool</td>
<td>true, false</td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_URL_DETECT_FILES</td>
<td>163</td>
<td>bool</td>
<td>false</td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_URL_UNDERLINE</td>
<td>166</td>
<td>bool</td>
<td>true, false</td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_URL_OVERWRITE_COLOR</td>
<td>167</td>
<td>bool</td>
<td>false</td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_URL_COLOR</td>
<td>168</td>
<td>long</td>
<td>RGB color</td>
<td></td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_URL_CHECK_FILE_EXISTS</td>
<td>164</td>
<td>bool</td>
<td>false</td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
</tbody>
</table>

### OVERLAY

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_OVERLAY_FILE</td>
<td>93</td>
<td>string</td>
<td>1 - Local, 2 - Server</td>
<td></td>
<td>SetOverlayOptionString</td>
</tr>
<tr>
<td>NOVAPDF_OVERLAY_FILE_LOCATION</td>
<td>96</td>
<td>long</td>
<td>1 - Local, 2 - Server</td>
<td>1</td>
<td>SetOverlayOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_OVERLAY_FILE_PASSWORD</td>
<td>97</td>
<td>string (encrypted)</td>
<td>password for the overlay PDF file, if encrypted</td>
<td></td>
<td>SetOverlayOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_OVERLAY_USER_NAME</td>
<td>94</td>
<td>string</td>
<td>user name for network path</td>
<td></td>
<td>SetOverlayOptionString</td>
</tr>
<tr>
<td>NOVAPDF_OVERLAY_USER_PASSWORD</td>
<td>95</td>
<td>string (encrypted)</td>
<td>user password</td>
<td></td>
<td>SetOverlayOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_OVERLAY_REPEAT_TYPE</td>
<td>99</td>
<td>long</td>
<td>0 - no repeat; 1 - repeat last page; 2 - repeat all pages</td>
<td>0</td>
<td>SetOverlayOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_OVERLAY_OPACITY</td>
<td>100</td>
<td>long</td>
<td>100</td>
<td></td>
<td>SetOverlayOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_OVERLAY_NAME</td>
<td>362</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOverlayOptionString</td>
</tr>
<tr>
<td>NOVAPDF_OVERLAY_AUTHOR</td>
<td>363</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOverlayOptionString</td>
</tr>
<tr>
<td>NOVAPDF_OVERLAY_DESCRIPTION</td>
<td>364</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOverlayOptionString</td>
</tr>
</tbody>
</table>

### IMAGE WATERMARKS

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_IMAGE_WATeR</td>
<td>5</td>
<td>bool</td>
<td>true, false</td>
<td></td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>Constant name</td>
<td>Value</td>
<td>Type</td>
<td>Possible values</td>
<td>Default</td>
<td>Function</td>
</tr>
<tr>
<td>---------------</td>
<td>-------</td>
<td>------------</td>
<td>-----------------</td>
<td>---------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_IMAGE_FILE_NAME</td>
<td>200</td>
<td>string</td>
<td></td>
<td></td>
<td>SetWatermarkImageOptionString</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_IMAGE_LOCATION</td>
<td>201</td>
<td>long</td>
<td>1 - Local 2 - Server</td>
<td>1</td>
<td>SetWatermarkImageOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_IMAGE_USER_NAME</td>
<td>202</td>
<td>string</td>
<td>user name for network path</td>
<td></td>
<td>SetWatermarkImageOptionString</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_IMAGE_USER_PASSWORD</td>
<td>203</td>
<td>string (encrypted)</td>
<td>user password</td>
<td></td>
<td>SetWatermarkImageOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_IMAGE_TRANSPARENCY</td>
<td>204</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetWatermarkImageOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_IMAGE_TRANSP_COLOR</td>
<td>205</td>
<td>long</td>
<td>RGB color</td>
<td>0</td>
<td>SetWatermarkImageOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_IMAGE_OPACITY</td>
<td>206</td>
<td>long</td>
<td>100</td>
<td></td>
<td>SetWatermarkImageOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_COLOR_VARIATION</td>
<td>207</td>
<td>long</td>
<td>0</td>
<td></td>
<td>SetWatermarkImageOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_VISIBILITY_VIEW</td>
<td>208</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetWatermarkImageOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_VISIBILITY_PRINT</td>
<td>209</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetWatermarkImageOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_VISIBILITY_EXPORT</td>
<td>210</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetWatermarkImageOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_NAME</td>
<td>211</td>
<td>string</td>
<td></td>
<td></td>
<td>SetWatermarkImageOptionString</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_AUTHOR</td>
<td>212</td>
<td>string</td>
<td></td>
<td></td>
<td>SetWatermarkImageOptionString</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_DESCRIPTION</td>
<td>213</td>
<td>string</td>
<td></td>
<td></td>
<td>SetWatermarkImageOptionString</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_WATERMARK_TEXT</td>
<td>240</td>
<td>string</td>
<td></td>
<td></td>
<td>SetWatermarkTextOptionString</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_FONT_NAME</td>
<td>241</td>
<td>string</td>
<td></td>
<td></td>
<td>SetWatermarkTextOptionString</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_FONT_TYPE</td>
<td>242</td>
<td>long</td>
<td>0 - TrueType</td>
<td>0</td>
<td>SetWatermarkTextOptionLong</td>
</tr>
<tr>
<td>Constant name</td>
<td>Value</td>
<td>Type</td>
<td>Possible values</td>
<td>Default</td>
<td>Function</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>-------</td>
<td>------</td>
<td>-----------------</td>
<td>---------</td>
<td>-------------------</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_FONT_SIZE</td>
<td>243</td>
<td>long</td>
<td></td>
<td></td>
<td>SetWatermarkTextOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_FONT_BOLD</td>
<td>244</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetWatermarkTextOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_FONT_ITALIC</td>
<td>245</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetWatermarkTextOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_FONT_OUTLINE</td>
<td>246</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetWatermarkTextOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_FONT_UNDERLINE</td>
<td>247</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetWatermarkTextOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_FONT_COLOR</td>
<td>248</td>
<td>long</td>
<td>RGB color</td>
<td>100</td>
<td>SetWatermarkTextOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_OPACITY</td>
<td>249</td>
<td>long</td>
<td>1-100</td>
<td>100</td>
<td>SetWatermarkTextOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_VISIBILITY_VIEW</td>
<td>250</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetWatermarkTextOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_VISIBILITY_PRINT</td>
<td>251</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetWatermarkTextOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_WTM_IMG_VISIBILITY_EXPORT</td>
<td>252</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetWatermarkTextOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_UNDERLINE_POSITION</td>
<td>253</td>
<td>float</td>
<td>0</td>
<td></td>
<td>SetWatermarkTextOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_UNDERLINE_THICKNESS</td>
<td>254</td>
<td>float</td>
<td>0</td>
<td></td>
<td>SetWatermarkTextOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_TEXT_CORRECTION</td>
<td>255</td>
<td>float</td>
<td>0</td>
<td></td>
<td>SetWatermarkTextOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_NAME</td>
<td>256</td>
<td>string</td>
<td></td>
<td></td>
<td>SetWatermarkTextOptionString</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_AUTHOR</td>
<td>257</td>
<td>string</td>
<td></td>
<td></td>
<td>SetWatermarkTextOptionString</td>
</tr>
<tr>
<td>NOVAPDF_WTM_TXT_DESCRIPTION</td>
<td>258</td>
<td>string</td>
<td></td>
<td></td>
<td>SetWatermarkTextOptionString</td>
</tr>
</tbody>
</table>

**BOOKMARKS**

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_BMARK_EN_AUTO_DET_BMARKS</td>
<td>12</td>
<td>int</td>
<td></td>
<td>false</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_BMK_ALLOW_MULTILI</td>
<td>280</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetOptionBool</td>
</tr>
</tbody>
</table>
### NE

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default value</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_BMK_MATCH_ALL_LEVELS</td>
<td>281</td>
<td>bool</td>
<td>true</td>
<td>SetOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_BMK_USE_LEVELS</td>
<td>282</td>
<td>int</td>
<td>3</td>
<td>SetOptionLong</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_BMK_OPEN_TO_LEVEL</td>
<td>283</td>
<td>int</td>
<td>1</td>
<td>SetOptionLong</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_BMK_ROOT_NAME</td>
<td>284</td>
<td>string</td>
<td></td>
<td>SetOptionString</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_BMK_ROOT_ENABLED</td>
<td>285</td>
<td>bool</td>
<td>false</td>
<td>SetOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_BMK_ROOT_BOLD</td>
<td>286</td>
<td>bool</td>
<td>false</td>
<td>SetOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_BMK_ROOT_ITALIC</td>
<td>287</td>
<td>bool</td>
<td>false</td>
<td>SetOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_BMK_ROOT_COLOR</td>
<td>288</td>
<td>int</td>
<td></td>
<td>SetOptionLong</td>
<td></td>
</tr>
</tbody>
</table>

### SAVE

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default value</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_SAVE_PROMPT_TYPE</td>
<td>101</td>
<td>long</td>
<td>0 - standard dialog; 1 - no dialog; 2 - simple OS dialog</td>
<td>0</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_SAVE_FOLDER_TYPE</td>
<td>260</td>
<td>long</td>
<td>1 - Printing applications's current folder; 2 - last folder; 3 - custom folder; 4 - User's My Documents folder</td>
<td>4</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_SAVE_FOLDER</td>
<td>103</td>
<td>string</td>
<td>save file name or a valid macro</td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SAVE_USER_NAME</td>
<td>105</td>
<td>string</td>
<td>network user name</td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SAVE_USER_PASSWORD</td>
<td>106</td>
<td>string (encrypted)</td>
<td>network user password</td>
<td></td>
<td>SetOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_SAVE_FILE_NAME</td>
<td>104</td>
<td>string</td>
<td>save file name or a valid macro</td>
<td>[N]</td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SAVE_FILEEXIST_ACTION</td>
<td>108</td>
<td>long</td>
<td>0 - prompt save as dialog; 1 - autonumber new; 2 - append date-time; 3 - overwrite; 4 - auto number existing files; 7 - don't save file</td>
<td>0</td>
<td>SetOptionLong</td>
</tr>
</tbody>
</table>
### LAYOUT

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_LAYOUT_LEFTANCHOR_USED</td>
<td>320</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_LEFTANCHOR_OFFSET</td>
<td>321</td>
<td>float</td>
<td>0</td>
<td></td>
<td>SetLayoutOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_LEFTANCHOR_USE_PERCENT</td>
<td>322</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_RIGHTANCHOR_USED</td>
<td>323</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_RIGHTANCHOR_OFFSET</td>
<td>324</td>
<td>float</td>
<td>0</td>
<td></td>
<td>SetLayoutOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_RIGHTANCHOR_USE_PERCENT</td>
<td>325</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_TOPANCHOR_USED</td>
<td>326</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_TOPANCHOR_OFFSET</td>
<td>327</td>
<td>float</td>
<td>0</td>
<td></td>
<td>SetLayoutOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_TOPANCHOR_USE_PERCENT</td>
<td>328</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_BOTTOMANCHOR_USED</td>
<td>329</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_BOTTOMANCHOR_OFFSET</td>
<td>330</td>
<td>float</td>
<td>0</td>
<td></td>
<td>SetLayoutOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_BOTTOMANCHOR_USE_PERCENT</td>
<td>331</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_VERTANCHOR_USED</td>
<td>332</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_VERTANCHOR_OFFSET</td>
<td>333</td>
<td>float</td>
<td>0</td>
<td></td>
<td>SetLayoutOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_VERTANCHOR_USE_PERCENT</td>
<td>334</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_HORIZANCHOR_USED</td>
<td>335</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_HORIZANCHOR_OFFSET</td>
<td>336</td>
<td>float</td>
<td>0</td>
<td></td>
<td>SetLayoutOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_HORIZANCHOR_USE_PERCENT</td>
<td>337</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_USE_ASPECT_RATIO</td>
<td>338</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetLayoutOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_WIDTH</td>
<td>339</td>
<td>float</td>
<td></td>
<td></td>
<td>SetLayoutOptionFloat</td>
</tr>
<tr>
<td>Constant</td>
<td>Type</td>
<td>Value</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------</td>
<td>------------------------</td>
<td>-------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_HEIGHT</td>
<td>float</td>
<td>340</td>
<td>SetLayoutOptionFloat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_ROTATION</td>
<td>float</td>
<td>341</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_UNITS</td>
<td>long</td>
<td>342</td>
<td>SetLayoutOptionLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_PAGERANGE</td>
<td>string</td>
<td>349</td>
<td>&quot;1-2&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_ZINDEX</td>
<td>long</td>
<td>350</td>
<td>multiple of 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_FORM_WIDTH</td>
<td>float</td>
<td>351</td>
<td>SetLayoutOptionFloat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_FORM_HEIGHT</td>
<td>float</td>
<td>352</td>
<td>SetLayoutOptionFloat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_BOUNDING_WIDTH</td>
<td>float</td>
<td>353</td>
<td>SetLayoutOptionFloat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_BOUNDING_HEIGHT</td>
<td>float</td>
<td>354</td>
<td>SetLayoutOptionFloat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUTORIGINAL_WIDTH</td>
<td>float</td>
<td>355</td>
<td>SetLayoutOptionFloat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUTORIGINAL_HEIGHT</td>
<td>float</td>
<td>356</td>
<td>SetLayoutOptionFloat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_SCALE_X</td>
<td>float</td>
<td>360</td>
<td>SetLayoutOptionFloat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_SCALE_Y</td>
<td>float</td>
<td>361</td>
<td>SetLayoutOptionFloat</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_ALLOW_STRETCH</td>
<td>bool</td>
<td>386</td>
<td>SetLayoutOptionBool</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_USE_BORDER</td>
<td>bool</td>
<td>380</td>
<td>SetLayoutOptionBool</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_BORDER_STYLE</td>
<td>long</td>
<td>382</td>
<td>SetLayoutOptionLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_BORDER_WIDTH</td>
<td>long</td>
<td>383</td>
<td>SetLayoutOptionLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_BORDER_COLOR</td>
<td>long</td>
<td>384</td>
<td>SetLayoutOptionLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_BORDER_POSITION</td>
<td>long</td>
<td>385</td>
<td>SetLayoutOptionLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_NAME</td>
<td>string</td>
<td>357</td>
<td>SetLayoutOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_AUTHOR</td>
<td>string</td>
<td>358</td>
<td>SetLayoutOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_DESCRIPTION</td>
<td>string</td>
<td>359</td>
<td>SetLayoutOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_ALLOW_STRETCH</td>
<td>bool</td>
<td>386</td>
<td>true</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_OBJECT_WIDTH</td>
<td>float</td>
<td>393</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_OBJECT_HEIGHT</td>
<td>float</td>
<td>394</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_PAGEVISIBILITY</td>
<td>long</td>
<td>397</td>
<td>0 - First</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## TY

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_LAYOUT_PAGE_FORM_HEIGHT</td>
<td>398</td>
<td>float</td>
<td></td>
<td>210000</td>
<td>SetLayoutOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_PAGE_FORM_WIDTH</td>
<td>399</td>
<td>float</td>
<td></td>
<td>297000</td>
<td>SetLayoutOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_LAYOUT_PAGE_FORM_ID</td>
<td>400</td>
<td>long</td>
<td>DMPAPER sizes</td>
<td>9 (A4)</td>
<td>SetLayoutOptionLong</td>
</tr>
</tbody>
</table>

## ADVANCED OPTIONS

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_ADV_OPTIMIZE_TEXT</td>
<td>48</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ADV_SILENT_PRINT</td>
<td>49</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ADV_OPTIMIZE_IMAGE</td>
<td>50</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ADV_CORRECT_LINEWIDTH</td>
<td>51</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ADV_CORRECT_FILLCOLOR</td>
<td>52</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ADV_IGNORE_EMPTY_PAGES</td>
<td>53</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ADV_VERIFY_FILE</td>
<td>621</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ADV_COPIES_ADD_PAGE</td>
<td>622</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ADV_COPIES_ONLY_ODD</td>
<td>623</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
</tbody>
</table>

## ACTIONS - RUN APPLICATION

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
</table>

 novaPDF SDK 11 User Manual  Copyright © 2021 Softland
<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_ACTION_RUN_APPLICATION</td>
<td>43</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_RUN_USER_NAME</td>
<td>45</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_RUN_USER_PASSWORD</td>
<td>46</td>
<td>string (encrypted)</td>
<td></td>
<td></td>
<td>SetActionOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_RUN_CMDLINE_PARAMETERS</td>
<td>47</td>
<td>string</td>
<td>%1 - pdf filename%2 - process name</td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_RUN_SHOW</td>
<td>640</td>
<td>long</td>
<td>0 - Normal1 - Minimized2 - Hidden</td>
<td></td>
<td>SetActionOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_RUN_SHELL</td>
<td>641</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetActionOptionBool</td>
</tr>
</tbody>
</table>

**ACTIONS - OPEN VIEWER**

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_ACTION_OPEN_CUSTOMSOURCE</td>
<td>460</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_OPEN_ORIGINAL</td>
<td>461</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_OPEN_USERNAME</td>
<td>462</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_OPEN_PASSWORD</td>
<td>463</td>
<td>string (encrypted)</td>
<td></td>
<td></td>
<td>SetActionOptionEncryptedString</td>
</tr>
</tbody>
</table>

**ACTIONS - DELETE**

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_ACTION_DELETE_CUSTOMSOURCE</td>
<td>464</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_DELETE_DELETE_ORIGINAL</td>
<td>465</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_DELETE_USERNAME</td>
<td>466</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
</tbody>
</table>
### ACTIONS - COPY

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_ACTION_COPY_CUSTOMSOURCE</td>
<td>468</td>
<td>string</td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_COPY_SOURCECOPYORIGINAL</td>
<td>469</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_COPY_DESTCOPYORIGINAL</td>
<td>470</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_COPY_CUSTOMDESTFOLDER</td>
<td>471</td>
<td>string</td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_COPY_CUSTOMDESTFILE</td>
<td>472</td>
<td>string</td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_COPY_SOURCEUSERNAME</td>
<td>473</td>
<td>string</td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_COPY_SOURCEPASSWORD</td>
<td>474</td>
<td>string (encrypted)</td>
<td></td>
<td>SetActionOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_COPY_DESTUSERNAME</td>
<td>475</td>
<td>string</td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_COPY_DESTPASSWORD</td>
<td>476</td>
<td>string (encrypted)</td>
<td></td>
<td>SetActionOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_COPY_DESTKEEPFILENAME</td>
<td>477</td>
<td>bool</td>
<td>false</td>
<td>SetActionOptionBool</td>
</tr>
</tbody>
</table>

### ACTIONS - EMAIL MAPI

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_EMAILMAPI_SENDEMAIL</td>
<td>421</td>
<td>long</td>
<td>0 - Send with default email client</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 - Open default email</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_FROM</td>
<td>422</td>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_TO</td>
<td>423</td>
<td>string</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### ACTIONS - EMAIL OUTLOOK

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_OUTLOOK_IMPORTANCE</td>
<td>645</td>
<td>long</td>
<td>0 - Low, 1 - Normal, 2 - High</td>
<td>1</td>
<td>SetActionOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_READRECEIPT</td>
<td>646</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_NOFORWARD</td>
<td>647</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_NOREPLY</td>
<td>648</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_NOREPLYALL</td>
<td>649</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_SHOWWINDOW</td>
<td>650</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_DELETEAFTERSEND</td>
<td>651</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOKCATEGORY</td>
<td>652</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
</tbody>
</table>

### EMAILMAPI Constants

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_EMAILMAPI_CC</td>
<td>424</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_BCC</td>
<td>425</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_COMPRESS</td>
<td>426</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_PASSWORDPROTECT</td>
<td>670</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_ZIPPASSWORD</td>
<td>671</td>
<td>string</td>
<td>(encrypted)</td>
<td></td>
<td>SetActionOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_CHANGEEXTENSION</td>
<td>427</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_ATTACHPDF</td>
<td>428</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_LOOKUPADDRESS</td>
<td>430</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_ATTACHOTHERFILES</td>
<td>431</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_OTHERFILES</td>
<td>432</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_SUBJECT</td>
<td>433</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_BODY</td>
<td>434</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILMAPI_EXTENSION</td>
<td>435</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>Constant name</td>
<td>Value</td>
<td>Type</td>
<td>Possible values</td>
<td>Default</td>
<td>Function</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------</td>
<td>-------------------</td>
<td>-----------------</td>
<td>---------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_PASSWORDPROTECT</td>
<td>653</td>
<td>bool</td>
<td>false</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_ZIPPASSWORD</td>
<td>654</td>
<td>string (encrypted)</td>
<td></td>
<td></td>
<td>SetActionOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_COMPRESS</td>
<td>655</td>
<td>bool</td>
<td>false</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_CHANGEEXTENSION</td>
<td>656</td>
<td>bool</td>
<td>false</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_ATTACHMENT_PDF</td>
<td>657</td>
<td>bool</td>
<td>false</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_LOOKUP_ADDRESS</td>
<td>658</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_ATTACHMENT_OTHER_FILES</td>
<td>659</td>
<td>bool</td>
<td>false</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_OTHER_FILES</td>
<td>660</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_SUBJECT</td>
<td>661</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_BODY</td>
<td>662</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_FROM</td>
<td>663</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_TO</td>
<td>664</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_CC</td>
<td>665</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_BCC</td>
<td>666</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_OUTLOOK_EXTENSION</td>
<td>667</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
</tbody>
</table>

### ACTIONS - EMAIL SMTP

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_EMAILSMTP_FROM</td>
<td>436</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_TO</td>
<td>437</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_CC</td>
<td>438</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_BCC</td>
<td>439</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_COMPRESS</td>
<td>440</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_PASSWORDPROTECT</td>
<td>668</td>
<td>bool</td>
<td></td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_ZIPPASSWORD</td>
<td>669</td>
<td>string (encrypted)</td>
<td></td>
<td></td>
<td>SetActionOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_CHANGEEXTENSION</td>
<td>441</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>Constant name</td>
<td>Value</td>
<td>Type</td>
<td>Possible values</td>
<td>Default</td>
<td>Function</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-------</td>
<td>----------</td>
<td>----------------------------------------</td>
<td>---------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_ATTACHMENT_PDF</td>
<td></td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_ATTACHMENT_OTHER_FILES</td>
<td></td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_OTHER_FILES</td>
<td></td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_SUBJECT</td>
<td></td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_BODY</td>
<td></td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_EXTENSION</td>
<td></td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_SERVER</td>
<td></td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_PORT</td>
<td></td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_SSL</td>
<td></td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_AUTHENTICATION</td>
<td></td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_ACCOUNT</td>
<td></td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_PASSWORD</td>
<td></td>
<td>string</td>
<td>encrypted</td>
<td></td>
<td>SetActionOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_TLS</td>
<td></td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILSMTP_SENDASTEXT</td>
<td></td>
<td>bool</td>
<td>true</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
</tbody>
</table>

**ACTIONS - EMAIL OAUTH (GMAIL)**

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_EMAILOAUTH_TYPE</td>
<td>700</td>
<td>long</td>
<td>0 - GMail</td>
<td>0</td>
<td>SetActionOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_EMAILOAUTH_ACCOUNT_USAGE</td>
<td>701</td>
<td>long</td>
<td>0 - Use one account 1 - ask account 2 - reuse account</td>
<td></td>
<td>SetActionOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_EMAILOAUTH_TO</td>
<td>702</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILOAUTH_CC</td>
<td>703</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILOAUTH_BCC</td>
<td>704</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILOAUTH_COMPRESS</td>
<td>705</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILOAUTH_PASSWORD_PROTECT</td>
<td>714</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>Constant name</td>
<td>Value</td>
<td>Type</td>
<td>Possible values</td>
<td>Default</td>
<td>Function</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-------</td>
<td>---------------</td>
<td>-----------------</td>
<td>---------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>NOVAPDF_EMAILAUTH_ZIPPASSWORD</td>
<td>715</td>
<td>string</td>
<td>(encrypted)</td>
<td></td>
<td>SetActionOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILAUTH_CHANGEEXTENSION</td>
<td>706</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILAUTH_ATTACH_PDF</td>
<td>707</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILAUTH_ATTACH_OTHER_FILES</td>
<td>708</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetActionOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_EMAILAUTH_OTHER_FILES</td>
<td>709</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILAUTH_SUBJECT</td>
<td>710</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILAUTH_BODY</td>
<td>711</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILAUTH_EXTENSION</td>
<td>712</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_EMAILAUTH_SENDASTEXT</td>
<td>713</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetActionOptionString</td>
</tr>
</tbody>
</table>

**ACTIONS - FTP**

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_ACTION_FTP_SOURCEFILE</td>
<td>480</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_SOURCEUSER</td>
<td>481</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_SOURCEPASSWORD</td>
<td>482</td>
<td>string</td>
<td>(encrypted)</td>
<td></td>
<td>SetActionOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_DESTFOLDER</td>
<td>483</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_DESTFILE</td>
<td>484</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_SERVER</td>
<td>485</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_PORT</td>
<td>486</td>
<td>string</td>
<td></td>
<td>21</td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_USERNAME</td>
<td>487</td>
<td>string</td>
<td></td>
<td></td>
<td>SetActionOptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_PASSWORD</td>
<td>488</td>
<td>string</td>
<td>(encrypted)</td>
<td></td>
<td>SetActionOptionEncryptedString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_UPLOAD</td>
<td>489</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetActionOptionBool</td>
</tr>
</tbody>
</table>
### ACTIONS - FTP

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_ACTION_FTP_UPLOAD_SPEED</td>
<td>490</td>
<td>long</td>
<td>200</td>
<td>SetActionOptionLong</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_USEPASSIVEMODE</td>
<td>491</td>
<td>bool</td>
<td>false</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_TIMEOUT</td>
<td>492</td>
<td>long</td>
<td>30</td>
<td>SetActionOptionLong</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_HOST</td>
<td>493</td>
<td>string</td>
<td></td>
<td>SetActionOptionString</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_HOSTPORT</td>
<td>494</td>
<td>string</td>
<td>80</td>
<td>SetActionOptionString</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_HOSTUSERNAME</td>
<td>495</td>
<td>string</td>
<td></td>
<td>SetActionOptionString</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_HOSTPASSWORD</td>
<td>496</td>
<td>string (encrypted)</td>
<td></td>
<td>SetActionOptionEncryptedString</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_RETRY</td>
<td>497</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_RETRYVALUE</td>
<td>498</td>
<td>long</td>
<td>1</td>
<td>SetActionOptionLong</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_WAIT</td>
<td>499</td>
<td>bool</td>
<td>false</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_WAITVALUE</td>
<td>500</td>
<td>long</td>
<td>60</td>
<td>SetActionOptionLong</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_SSL</td>
<td>501</td>
<td>bool</td>
<td>false</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_USEPROXY</td>
<td>502</td>
<td>bool</td>
<td>false</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_SOURCEORIGINAL</td>
<td>503</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FTP_KEEPFILENAME</td>
<td>504</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
</tbody>
</table>

### ACTIONS - SFTP

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_ACTION_SFTP_SOURCEORIGINAL</td>
<td>510</td>
<td>bool</td>
<td>true</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_SOURCEFILE</td>
<td>511</td>
<td>string</td>
<td></td>
<td>SetActionOptionString</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_SOURCEUSER</td>
<td>512</td>
<td>string</td>
<td></td>
<td>SetActionOptionString</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_SOURCEPASSWORD</td>
<td>513</td>
<td>string (encrypted)</td>
<td></td>
<td>SetActionOptionEncryptedString</td>
<td></td>
</tr>
<tr>
<td>Path Reference</td>
<td>Value</td>
<td>Type</td>
<td>Method</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-------</td>
<td>--------------</td>
<td>---------------------------------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_DESTFOLDER</td>
<td></td>
<td>string</td>
<td>SetActionOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_DESTFILE</td>
<td></td>
<td>string</td>
<td>SetActionOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_SERVER</td>
<td></td>
<td>string</td>
<td>SetActionOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_PORT</td>
<td></td>
<td>string</td>
<td>SetActionOptionString</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_USERNAME</td>
<td></td>
<td>string</td>
<td>SetActionOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_PASSWORD</td>
<td></td>
<td>string (encrypted)</td>
<td>SetActionOptionEncryptedString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_USEPRIVATEKEY</td>
<td></td>
<td>bool</td>
<td>SetActionOptionBool</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_UPLOADSPEEDCHECK</td>
<td></td>
<td>bool</td>
<td>SetActionOptionBool</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_UPLOADSPEED</td>
<td></td>
<td>long</td>
<td>SetActionOptionLong</td>
<td>200</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_TIMEOUT</td>
<td></td>
<td>long</td>
<td>SetActionOptionLong</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_KEYFILENAME</td>
<td></td>
<td>string</td>
<td>SetActionOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_KEYFILEPASSWORD</td>
<td></td>
<td>string (encrypted)</td>
<td>SetActionOptionEncryptedString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_KEEPFILENAME</td>
<td></td>
<td>string</td>
<td>SetActionOptionString</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_RETRY</td>
<td></td>
<td>bool</td>
<td>SetActionOptionBool</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_RETRYVALUE</td>
<td></td>
<td>long</td>
<td>SetActionOptionLong</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_WAIT</td>
<td></td>
<td>bool</td>
<td>SetActionOptionBool</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_WAITVALUE</td>
<td></td>
<td>long</td>
<td>SetActionOptionLong</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_AUTH_PUBLICKEY</td>
<td></td>
<td>bool</td>
<td>SetActionOptionBool</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_AUTH_PRIVATEKEY</td>
<td></td>
<td>bool</td>
<td>SetActionOptionBool</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_AUTH_KEYBOARDINT</td>
<td></td>
<td>bool</td>
<td>SetActionOptionBool</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_AES256CTR</td>
<td></td>
<td>bool</td>
<td>SetActionOptionBool</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_3DESCBC</td>
<td></td>
<td>bool</td>
<td>SetActionOptionBool</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_AES1_28CBC</td>
<td></td>
<td>bool</td>
<td>SetActionOptionBool</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Value</td>
<td>Type</td>
<td>Default</td>
<td>Method</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>-------</td>
<td>-------</td>
<td>---------</td>
<td>-----------------</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_AES128_CBC</td>
<td>553</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_AES192_CBC</td>
<td>554</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_TWOFISH128_CBC</td>
<td>557</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_TWOFISH192_CBC</td>
<td>558</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_TWOFISH256_CBC</td>
<td>559</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_ARCFOUR128_CBC</td>
<td>560</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_ARCFOUR256_CBC</td>
<td>561</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_CAST128_CBC</td>
<td>562</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_AES128_CTR</td>
<td>564</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_CIPHER_AES192_CTR</td>
<td>565</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_MAC_MD5</td>
<td>570</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_MAC_SHA1</td>
<td>571</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_MAC_SHA225_64</td>
<td>572</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_MAC_SHA225_696</td>
<td>573</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_MAC_RIPEMD160</td>
<td>574</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_MAC_RIPEMD160_OPENSSH</td>
<td>575</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_MAC_MD596</td>
<td>576</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SFTP_MAC_SHA192</td>
<td>577</td>
<td>bool</td>
<td>true</td>
<td>SetActionOptionBool</td>
<td></td>
</tr>
</tbody>
</table>
### GENERAL ACTION SETTINGS

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_ACTION_INDEX</td>
<td>600</td>
<td>long</td>
<td></td>
<td></td>
<td>SetActionOptions</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_FAIL</td>
<td>601</td>
<td>long</td>
<td>0 - Continue 1 - Stop</td>
<td>1</td>
<td>ActionOptions</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_NAME</td>
<td>602</td>
<td>string</td>
<td></td>
<td></td>
<td>OptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_SHOW_DIALOG</td>
<td>603</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>OptionString</td>
</tr>
<tr>
<td>NOVAPDF_ACTION_EXECUTE</td>
<td>604</td>
<td>bool</td>
<td>true</td>
<td></td>
<td>OptionString</td>
</tr>
</tbody>
</table>

### XMP METADATA

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_DOCINFO_METADATA</td>
<td>618</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>OptionString</td>
</tr>
<tr>
<td>NOVAPDF_COPYRIGHT_METADATA</td>
<td>17</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>OptionString</td>
</tr>
<tr>
<td>NOVAPDF_COPYRIGHT_TEXT</td>
<td>611</td>
<td>string</td>
<td></td>
<td></td>
<td>OptionString</td>
</tr>
<tr>
<td>NOVAPDF_COPYRIGHT_URL</td>
<td>612</td>
<td>string</td>
<td></td>
<td></td>
<td>OptionString</td>
</tr>
<tr>
<td>NOVAPDF_COPYRIGHT_TYPE</td>
<td>613</td>
<td>long</td>
<td>0 - Unknown 1 - Copyrighted 2 - Public domain</td>
<td>0</td>
<td>OptionLong</td>
</tr>
<tr>
<td>NOVAPDF_CUSTOMPROP_METADATA</td>
<td>20</td>
<td>bool</td>
<td>false</td>
<td></td>
<td>OptionString</td>
</tr>
<tr>
<td>Constant name</td>
<td>Value</td>
<td>Type</td>
<td>Possible values</td>
<td>Default</td>
<td>Function</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>-------</td>
<td>---------</td>
<td>------------------------------------------------------</td>
<td>---------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>NOVAPDF_PROFILE_NAME</td>
<td>22</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_PROFILE_AUTHOR</td>
<td>23</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_PROFILE_DESCRIPTION</td>
<td>24</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SAVE_LOCATION</td>
<td>14</td>
<td>long</td>
<td>1 - Local</td>
<td>1</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_PROFILE_NAME</td>
<td>22</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_PROFILE_AUTHOR</td>
<td>23</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_PROFILE_DESCRIPTION</td>
<td>24</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SAVE_LOCATION</td>
<td>14</td>
<td>long</td>
<td>1 - Local</td>
<td>1</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_PROFILE_NAME</td>
<td>22</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_PROFILE_AUTHOR</td>
<td>23</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_PROFILE_DESCRIPTION</td>
<td>24</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_SAVE_LOCATION</td>
<td>14</td>
<td>long</td>
<td>1 - Local</td>
<td>1</td>
<td>SetOptionLong</td>
</tr>
</tbody>
</table>
### SetOptionBool

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_PDF_A</td>
<td>2</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_PDF_A_VERSION</td>
<td>3</td>
<td>long</td>
<td>1 - PDFA\1a</td>
<td>2</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 - PDFA\1b</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 - PDFA\2a</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4 - PDFA\2b</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5 - PDFA\2u</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6 - PDFA\3a</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7 - PDFA\3b</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8 - PDFA\3u</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_PDF_LINEARIZE</td>
<td>4</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_ENABLE_SAVE_RULES</td>
<td>690</td>
<td>bool</td>
<td></td>
<td>true</td>
<td>SetOptionBool</td>
</tr>
</tbody>
</table>

### SetOptionLong

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_ENABLE_USER_TAGS</td>
<td>21</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_REMOVE_POLICY</td>
<td>672</td>
<td>long</td>
<td></td>
<td></td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_DETECT_FONTNAME</td>
<td>673</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_FONTNAME</td>
<td>674</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_FONTTYPE</td>
<td>675</td>
<td>long</td>
<td>0 - TrueType</td>
<td>0</td>
<td>SetOptionLong</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 - Type1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 - OpenType(TrueType)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 - OpenType(Tpe1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_DETECT_SIZE</td>
<td>676</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_SIZE</td>
<td>677</td>
<td>long</td>
<td></td>
<td></td>
<td>SetOptionLong</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_MARGIN</td>
<td>678</td>
<td>float</td>
<td></td>
<td></td>
<td>SetOptionFloat</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_DETECT_STYLE</td>
<td>679</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_BOLD</td>
<td>680</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_ITALIC</td>
<td>681</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_DETECT_COLOR</td>
<td>682</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_COLOR</td>
<td>683</td>
<td>long</td>
<td>RGB color</td>
<td></td>
<td>SetOptionLong</td>
</tr>
</tbody>
</table>

### SetOptionString

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_USERTAG_FONTNAME</td>
<td>674</td>
<td>string</td>
<td></td>
<td></td>
<td>SetOptionString</td>
</tr>
</tbody>
</table>

### SetOptionFloat

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_USERTAG_MARGIN</td>
<td>678</td>
<td>float</td>
<td></td>
<td></td>
<td>SetOptionFloat</td>
</tr>
</tbody>
</table>

### SetOption

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_USERTAG_BOLD</td>
<td>680</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_ITALIC</td>
<td>681</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_DETECT_COLOR</td>
<td>682</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_USERTAG_COLOR</td>
<td>683</td>
<td>long</td>
<td>RGB color</td>
<td></td>
<td>SetOptionLong</td>
</tr>
</tbody>
</table>
## NUP

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_NUP_ENABLE</td>
<td>720</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
<tr>
<td>NOVAPDF_NUP_SHOW_DLG</td>
<td>721</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetOptionBool</td>
</tr>
</tbody>
</table>
| NOVAPDF_NUP_SHEET_SIZE_TYPE| 722   | long   | 0 - Auto (from printed document)  
1 - Predefined (NOVAPDF_NUP_SHEET_SIZE)  
2 - Custom (set width and height) | 0       | SetOptionLong                   |
| NOVAPDF_NUP_SHEET_SIZE     | 723   | long   | Windows paper size like DMPAPER_A4                                            | 0       | SetOptionLong     |
| NOVAPDF_NUP_SHEET_WIDTH    | 724   | long   | mm * 100                                                                       | 0       | SetOptionLong     |
| NOVAPDF_NUP_SHEET_HEIGHT   | 725   | long   | mm * 100                                                                       | 0       | SetOptionLong     |
| NOVAPDF_NUP_PAGE_ORDER     | 726   | long   | 0 - Across Left  
1 - Across Right  
2 - Down Left  
3 - Down Right | 5       | SetOptionLong                   |
| NOVAPDF_NUP_PAGE_PER_SHEET | 727   | long   | 0 - 1 page  
1 - 2 pages  
2 - 4 pages  
3 - 6 pages  
4 - 8 pages  
5 - 9 pages  
6 - 16 pages | 1       | SetOptionLong                   |

## PRINTER OPTIONS

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF_PRINTER_SELECT_PROFILE_DLG</td>
<td>802</td>
<td>bool</td>
<td></td>
<td>false</td>
<td>SetPrinterOption</td>
</tr>
</tbody>
</table>

## OVERRIDE OPTIONS

<table>
<thead>
<tr>
<th>Constant name</th>
<th>Value</th>
<th>Type</th>
<th>Possible values</th>
<th>Default</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant Name</td>
<td>Value</td>
<td>Type</td>
<td>Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>--------</td>
<td>---------------</td>
<td>---------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_OPEN_VIEWER</td>
<td>2001</td>
<td>bool</td>
<td>SetOverrideOptionBool</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_SEND_EMAIL</td>
<td>2002</td>
<td>bool</td>
<td>SetOverrideOptionBool</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_MERGE_PDF</td>
<td>2003</td>
<td>bool</td>
<td>SetOverrideOptionBool</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_MERGE_FILE_TYPE</td>
<td>2004</td>
<td>long</td>
<td>SetOverrideOptionLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_MERGE_TYPE</td>
<td>2005</td>
<td>long</td>
<td>SetOverrideOptionLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_MERGE_FILE</td>
<td>2006</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_INSERT_PAGENO</td>
<td>2007</td>
<td>long</td>
<td>SetOverrideOptionLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_PASSWORD</td>
<td>2008</td>
<td>string (enc)</td>
<td>SetOverrideOptionEncryptedString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_DOC_INFO</td>
<td>2009</td>
<td>bool</td>
<td>SetOverrideOptionBool</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_SECURITY</td>
<td>2010</td>
<td>bool</td>
<td>SetOverrideOptionBool</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_SECURITY_U</td>
<td>2011</td>
<td>string (enc)</td>
<td>SetOverrideOptionEncryptedString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_SECURITY_O</td>
<td>2012</td>
<td>string (enc)</td>
<td>SetOverrideOptionEncryptedString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_DOC_TITLE</td>
<td>2013</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_DOC_SUBJECT</td>
<td>2014</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_DOC_AUTHOR</td>
<td>2015</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_DOC_KEY</td>
<td>2016</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_DOC_CREATOR</td>
<td>2017</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_REC_TO</td>
<td>2018</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_REC_FROM</td>
<td>2019</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>------</td>
<td>--------</td>
<td>-------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_REC_CC</td>
<td>2020</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_REC_BCC</td>
<td>2021</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_EMBEDFONTS</td>
<td>2022</td>
<td>bool</td>
<td>SetOverrideOptionBool</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_COMPRESSION</td>
<td>2023</td>
<td>long</td>
<td>SetOverrideOptionLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_SAVE_TYPE</td>
<td>2024</td>
<td>long</td>
<td>SetOverrideOptionLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_SAVE_FOLDER</td>
<td>2025</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_SAVE_FILE</td>
<td>2026</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_SAVE_USER</td>
<td>2027</td>
<td>string</td>
<td>SetOverrideOptionString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_SAVE_PASSWORD</td>
<td>2028</td>
<td>string (encrypted)</td>
<td>SetOverrideOptionEncryptedString</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_SAVE_CONFLICT</td>
<td>2029</td>
<td>long</td>
<td>SetOverrideOptionLong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOVAPDF_OVERRIDE_SAVE_FOLDER_TYPE</td>
<td>2030</td>
<td>long</td>
<td>SetOverrideOptionLong</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Windows messages

**novaPDF SDK 11 messages**

<table>
<thead>
<tr>
<th>Message name</th>
<th>Event</th>
<th>WPARAM</th>
<th>LPARAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOVAPDF2_STARTDOC</td>
<td>sent when the printer driver begins processing the print job and generating the PDF file</td>
<td>0</td>
<td>jobID</td>
</tr>
<tr>
<td>NOVAPDF2_ENDDOC</td>
<td>sent when the printer driver finished processing the print job</td>
<td>0</td>
<td>jobID</td>
</tr>
<tr>
<td>NOVAPDF2_STARTPAGE</td>
<td>sent when the printer driver starts processing a new page</td>
<td>0</td>
<td>jobID</td>
</tr>
<tr>
<td>NOVAPDF2_ENDPAGE</td>
<td>sent when the printer driver finished processing a page</td>
<td>0</td>
<td>jobID</td>
</tr>
<tr>
<td>NOVAPDF2_FILESENT</td>
<td>sent when the printer driver finished generating the PDF file and sent it to the computer that started the print job</td>
<td>0</td>
<td>jobID</td>
</tr>
<tr>
<td>NOVAPDF2_FILESAVED</td>
<td>sent when the PDF file is received and saved by the computer that started the print job</td>
<td>0</td>
<td>jobID</td>
</tr>
<tr>
<td>NOVAPDF2_PRINTERROR</td>
<td>sent when an error occurred during the print job</td>
<td>error no.</td>
<td>jobID</td>
</tr>
<tr>
<td>NOVAPDF2_EMAILSENT</td>
<td>sent when the email option is enabled and a email with the generated PDF file was sent</td>
<td>0</td>
<td>jobID</td>
</tr>
<tr>
<td>NOVAPDF2_EMAILERROR</td>
<td>sent when the email option is enabled and there was an error sending the email with the generated PDF file</td>
<td>0</td>
<td>jobID</td>
</tr>
</tbody>
</table>

The following error numbers are sent by the printer driver, in the NOVAPDF2_PRINTERROR event:

1. Error saving temporary PDF file on the printer server.
2. Error reading license information on the printer server.
3. Error generating the PDF file.
4. Print job was canceled.
5. Licensing error: too many copies running with the same license.
6. Client computer is not licensed.
7. Error sending email.
8. Could not read printer info.
9. Could not read profile.
10. Append pdf file - no or wrong password.
11. Insert before pdf file - no or wrong password.
14 - insert before pdf file - could not read file
15 - overlay pdf file - no or wrong password
16 - overlay pdf file - could not read file
18 - sign PDF - error signing
19 - sign PDF - error creating signature form
20 - sign PDF - image file not found
21 - image watermark - error drawing
22 - overlay - error drawing
25 - license error
26 - wrong parameters for action
27 - error creating folder
28 - error open viewer
29 - error run application
30 - file name error
31 - error copy file
32 - error delete file
33 - MAPI error
34 - SMTP error
35 - profile consistency error
36 - FTP error
37 - SFTP error
38 - error open file
39 - error pdf viewer
40 - access rights error
41 - open pdf error

How to register windows messages
You can register windows messages using RegisterWindowMessage function. Here are some samples of how to do it:
MFC Converter
VCL Converter

1.4.11.3 What is INovaPdfOptions
The INovaPdfOptions interface represents a COM object that allows the developers to set printing parameters for novaPDF SDK 11. This interface is derived from IDispatch interface directly.

This interface resides in the "novapi11.dll" module, that is distributed with the novaPDF SDK and is registered at install time.

INovaPdfOptions has next methods:

Initialization
Initialize
Initialize2
InitializeSilent
InitializeSilent2
Profiles Management
LoadProfile
LoadProfile2
SaveProfile
AddProfile
AddProfile2
CopyProfile
CopyProfile2
DeleteProfile
DeleteProfile2
GetFirstProfile
GetFirstProfile2
GetNextProfile
GetNextProfile2
GetActiveProfile
GetActiveProfile2
SetActiveProfile
SetActiveProfile2
SetPrinterActivePublicProfile
SetPrinterActivePublicProfile2
DeletePrinterActivePublicProfile
DeletePrinterActivePublicProfile2
SetPrinterPublicProfile
SetPrinterPublicProfile2
SetPrinterServerFlags
SetPrinterServerFlags2

Get / Set Profile Options
GetOptionString
GetOptionString2
SetOptionString
SetOptionString2
GetOptionLong
SetOptionLong
SetOptionBool
GetOptionBool
SetOptionEncryptedString
SetOptionEncryptedString2
GetOptionEncryptedString
GetOptionEncryptedString2

Actions options
AddAction
AddAction2
DeleteAction
DeleteAction2
GetActionCount
GetAction
GetAction2
GetActionOptionBool
GetActionOptionBool2
GetActionOptionEncryptedString
GetActionOptionEncryptedString2
GetActionOptionFloat
GetActionOptionFloat2
GetActionOptionLong
GetActionOptionLong2
GetActionOptionString
GetActionOptionString2
SetActionOptionBool
SetActionOptionBool2
SetActionOptionEncryptedString
SetActionOptionEncryptedString2
SetActionOptionFloat
SetActionOptionFloat2
SetActionOptionLong
SetActionOptionLong2
SetActionOptionString
SetActionOptionString2

Fonts options:
GetFontOption
GetFontOption2
SetFontOption
SetFontOption2
ClearFontOptions

Bookmarks options:
AddBookmarkDefinition
AddBookmarkDefinition2
ModifyBookmarkDefinition
ModifyBookmarkDefinition2
DeleteBookmarkDefinition
GetBookmarkDefinition
GetBookmarkDefinition2
GetBookmarkDefinitionCount

[*][*]
Image watermarks options:
AddWatermarkImage
AddWatermarkImage2
DeleteWatermarkImage
DeleteWatermarkImage2
GetWatermarkImage
GetWatermarkImage2
GetWatermarkImageCount
SetWatermarkImageOptionString
SetWatermarkImageOptionString2
SetWatermarkImageOptionLong
SetWatermarkImageOptionLong2
SetWatermarkImageOptionBool
SetWatermarkImageOptionBool2
SetWatermarkImageOptionFloat
SetWatermarkImageOptionFloat2
GetWatermarkImageOptionString
GetWatermarkImageOptionString2
GetWatermarkImageOptionLong
GetWatermarkImageOptionLong2
GetWatermarkImageOptionBool
GetWatermarkImageOptionBool2
GetWatermarkImageOptionFloat
GetWatermarkImageOptionFloat2
SetWatermarkImageOptionEncryptedString
SetWatermarkImageOptionEncryptedString2
GetWatermarkImageOptionEncryptedString
GetWatermarkImageOptionEncryptedString2

Text watermarks options:
AddWatermarkText
AddWatermarkText2
DeleteWatermarkText
DeleteWatermarkText2
GetWatermarkText
GetWatermarkText2
GetWatermarkTextCount
SetWatermarkTextOptionString
SetWatermarkTextOptionString2
SetWatermarkTextOptionLong
SetWatermarkTextOptionLong2
SetWatermarkTextOptionBool
SetWatermarkTextOptionBool2
SetWatermarkTextOptionFloat
SetWatermarkTextOptionFloat2
GetWatermarkTextOptionString
GetWatermarkTextOptionString2
GetWatermarkTextOptionLong
GetWatermarkTextOptionLong2
GetWatermarkTextOptionBool
GetWatermarkTextOptionBool2
GetWatermarkTextOptionFloat
GetWatermarkTextOptionFloat2

Layout options:
SetLayoutOptionString
SetLayoutOptionString2
SetLayoutOptionLong
SetLayoutOptionLong2
SetLayoutOptionBool
SetLayoutOptionBool2
SetLayoutOptionFloat
SetLayoutOptionFloat2
GetLayoutOptionString
GetLayoutOptionString2
GetLayoutOptionLong
GetLayoutOptionLong2
GetLayoutOptionBool
GetLayoutOptionFloat
GetLayoutOptionFloat2
GetLayoutCount
GetLayoutCount2
GetLayout
GetLayout2
DeleteLayout
DeleteLayout2

Signature options:
GetSignature
GetSignature2

Overlay options:
AddOverlay
AddOverlay2
GetOverlay
GetOverlay2
DeleteOverlay
DeleteOverlay2
GetOverlayCount
SetOverlayOptionBool
SetOverlayOptionBool2
SetOverlayOptionEncryptedString
SetOverlayOptionEncryptedString2
SetOverlayOptionFloat
SetOverlayOptionFloat2
SetOverlayOptionLong
SetOverlayOptionLong2
SetOverlayOptionString
SetOverlayOptionString2
GetOverlayOptionBool
GetOverlayOptionBool2
GetOverlayOptionFloat
GetOverlayOptionFloat2
GetOverlayOptionLong
GetOverlayOptionLong2
GetOverlayOptionString
GetOverlayOptionString2
GetOverlayOptionEncryptedString
GetOverlayOptionEncryptedString2

Content options:
GetContentLayout
GetContentLayout2

Custom properties:
AddCustomProperty
AddCustomProperty2
SetCustomProperty
SetCustomProperty2
GetCustomProperty
GetCustomProperty2
DeleteCustomProperty
DisableCustomProperty
EnableCustomProperty
GetCustomPropertiesCount

User tags:
AddUserTag
AddUserTag2
SetUserTag
SetUserTag2
GetUserTag
GetUserTag2
DeleteUserTag
DisableUserTag
EnableUserTag
ClearUserTags
GetUserTagsCount

Save rules:
AddSaveRule
AddSaveRule2
GetSaveRule
GetSaveRule2
SetSaveRule
SetSaveRule2
DeleteSaveRule
DisableSaveRule
EnableSaveRule
ClearSaveRules
GetSaveRulesCount

Printers management
AddNovaPrinter
AddNovaPrinter2
AddNovaPrinterExt
AddNovaPrinterExt2
DeleteNovaPrinter
DeleteNovaPrinterExt2
SetPrinterOption
SetDefaultPrinter
RestoreDefaultPrinter

Register events or messages
RegisterEventWindow
UnRegisterEventWindow
RegisterNovaEvent
RegisterNovaEvent2
WaitForNovaEvent
OLE Licensing
InitializeOLEUsage
LicenseOLEServer

ShellExecute Licensing
LicenseShellExecuteFile

Print from launched applications
LicenseApplication

Convert Office documents
ConvertWordDocument
ConvertWordDocument2
ConvertPowerPointDocument
ConvertPowerPointDocument2
ConvertPublisherDocument
ConvertPublisherDocument2
ConvertVisioDocument
ConvertVisioDocument2
ConvertExcelDocument
ConvertExcelDocument2

Override options
SetOverrideOptionBool
SetOverrideOptionEncryptedString
SetOverrideOptionEncryptedString2
SetOverrideOptionLong
SetOverrideOptionString
SetOverrideOptionString2
GetOverrideOptionBool
GetOverrideOptionEncryptedString
GetOverrideOptionEncryptedString2
GetOverrideOptionLong
GetOverrideOptionString
GetOverrideOptionString2
DeleteOverrideOptions

1.4.11.4 INovaPdfOptions
1.4.11.4.1 AddAction

The **AddAction** method adds an action

```c
HRESULT AddAction(
    [in]  LONG p_nType,
    [in, string] LPCWSTR p_wsActionName,
    [out, string] LPWSTR* p_pwsNewActionId,
    [out]  LONG* p_pnIndex
);```


Parameters:

- p_nType
  - [in] action type (1-copy, 2-delete, 3-MAPI email, 4-SMTP email, 6-open, 7-run application)
- p_wsActionName
  - [in] action name
- p_pwsNewActionId
  - [out] action id
- p_pnIndex
  - [out] action index

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong action type

Remarks:

This method adds a new action at the end of the list of actions. After adding an action, you can change action position in the list and set specific options for each action type.

### 1.4.11.4.2 AddAction2

The **AddAction2** method adds an action

```c
HRESULT AddAction2(
    [in] LONG p_nType,
    [in, string] BSTR p_wsActionName,
    [out, string] BSTR* p_pwsNewActionId,
    [out] LONG* p_pnIndex
);
```

Parameters:

- p_nType
  - [in] action type (1-copy, 2-delete, 3-MAPI email, 4-SMTP email, 6-open, 7-run application)
- p_wsActionName
  - [in] action name
- p_pwsNewActionId
  - [out] action id
- p_pnIndex
  - [out] action index

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong action type

Remarks:

This method adds a new action at the end of the list of actions. After adding an action, you can change action position in the list and set specific options for each action type.
1.4.11.4.3 AddBookmarkDefinition

The AddBookmarkDefinition method adds a new bookmark definition in the current loaded profile, having the characteristics specified by the method parameters.

```csharp
HRESULT AddBookmarkDefinition(
    [in] SHORT p_nHeading,
    [in] BOOL p_bEnabled,
    [in] BOOL p_bDetFont,
    [in, string] LPCWSTR p_wsDetFont,
    [in] LONG p_nDetFontType,
    [in] BOOL p_bDetStyle,
    [in] BOOL p_bDetBold,
    [in] BOOL p_bDetItalic,
    [in] BOOL p_bDetSize,
    [in] INT p_nDetSizeVal,
    [in] FLOAT p_nDetSizePt,
    [in] BOOL p_bDetColor,
    [in] LONG p_nDetColor,
    [in] BOOL p_bDispAsBold,
    [in] BOOL p_bDispAsItalic,
    [in] LONG p_nDispColor,
    [out] SHORT* p_nDefinition);
```

**Parameters:**

- `p_nHeading`  
  - [in] heading index where to add the definition (1-9)
- `p_bEnabled`  
  - [in] definition is enabled
- `p_bDetFont`  
  - [in] detect font
- `p_wsDetFont`  
  - [in] font name
- `p_nDetFontType`  
  - [in] font type: true type, type1, open type
- `p_bDetStyle`  
  - [in] detect font style
- `p_bDetBold`  
  - [in] bold font
- `p_bDetItalic`  
  - [in] italic font
- `p_bDetSize`  
  - [in] detect font size
- `p_nDetSizeVal`  
  - [in] font size
- `p_nDetSizePt`  
  - [in] font size rounding
- `p_bDetColor`  
  - [in] detect font color
- `p_nDetColor`  
  - [in] font color (RGB value)
- `p_bDispAsBold`  
  - [in] display bookmark font bold
- `p_bDispAsItalic`  
  - [in] display bookmark font italic
Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded

Remarks:
There can be defined maximum 9 bookmark definitions.

1.4.11.4.4 AddBookmarkDefinition2

The AddBookmarkDefinition2 method adds a new bookmark definition in the current loaded profile, having the characteristics specified by the method parameters.

```c
HRESULT AddBookmarkDefinition2(
    [in] SHORT p_nHeading,
    [in] BOOL p_bEnabled,
    [in] BOOL p_bDetFont,
    [in, string] BSTR p_wsDetFont,
    [in] LONG p_nDetFontType,
    [in] BOOL p_bDetStyle,
    [in] BOOL p_bDetBold,
    [in] BOOL p_bDetItalic,
    [in] BOOL p_bDetSize,
    [in] INT p_nDetSizeVal,
    [in] FLOAT p_nDetSizePt,
    [in] BOOL p_bDetColor,
    [in] LONG p_nDetColor,
    [in] BOOL p_bDispAsBold,
    [in] BOOL p_bDispAsItalic,
    [in] LONG p_nDispColor,
    [out] SHORT* p_nDefinition
);
```

Parameters:
- `p_nHeading` [in] heading index where to add the definition (1-9)
- `p_bEnabled` [in] definition is enabled
- `p_bDetFont` [in] detect font
- `p_wsDetFont` [in] font name
- `p_nDetFontType` [in] font type: true type, type1, open type
- `p_bDetStyle` [in] detect font style
- `p_bDetBold` [in] bold font
- `p_bDetItalic` [in] italic font
p_bDetSize
  [in] detect font size
p_nDetSizeVal
  [in] font size
p_nDetSizePt
  [in] font size rounding
p_bDetColor
  [in] detect font color
p_nDetColor
  [in] font color (RGB value)
p_bDispAsBold
  [in] display bookmark font bold
p_bDispAsItalic
  [in] display bookmark font italic
p_nDispColor
  [in] display bookmark font color
p_nDefinition
  [out] definition index, if added. If the definition was not added, -1

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_UNKNOWN_PROFILE - no profile loaded

Remarks:
  There can be defined maximum 9 bookmark definitions.

1.4.11.4.5 AddCustomProperty

The AddCustomProperty method adds a custom property

HRESULT AddCustomProperty(
    [in, string] LPCWSTR p_wsPropertyName,
    [in, string] LPCWSTR p_wsDefaultValue,
    [in] BOOL p_bEnabled,
    [out] LONG* p_pnIndex
);

Parameters:
  p_wsPropertyName
    [in] property name
  p_wsDefaultValue
    [in] property value
  p_bEnabled
    [in] flag, property is enabled
  p_pnIndex
    [out] custom property index

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_UNKNOWN_PROFILE - no profile loaded
  NV_INVALID_OPTION - empty property name
Remarks:
This method adds a new custom property at the end of the list.

### 1.4.11.4.6 AddCustomProperty2

The **AddCustomProperty2** method adds a custom property

```c
HRESULT AddCustomProperty2(
    [in, string] BSTR p_wsPropertyName,
    [in, string] BSTR p_wsPropertyValue,
    [in] BOOL p_bEnabled,
    [out] LONG* p_pnIndex
);
```

**Parameters:**
- `p_wsPropertyName` [in] property name
- `p_wsDefaultValue` [in] property value
- `p_bEnabled` [in] flag, property is enabled
- `p_pnIndex` [out] custom property index

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - empty property name

Remarks:
This method adds a new custom property at the end of the list.

### 1.4.11.4.7 AddNovaPrinter

The **AddNovaPrinter** method adds a temporary novaPDF printer in the system

```c
HRESULT AddNovaPrinter(
    [in] LPCWSTR p_wsPrinterName,
    [in, string] LPCWSTR p_wsOEMID,
    [in, string] LPCWSTR p_wsServicePort,
    [in, string] LPCWSTR p_wsLicensekey
);
```

**Parameters:**
- `p_wsPrinterName` [in] pointer to a null terminated Unicode string containing the name of the printer
- `p_wsOEMID` [in] custom OEMID; for trial is "nPdfSdk10_Softland"
- `p_wsServicePort` [in] novaPDF Server service port number,
p_wsLicenseKey
[in] license key

Return values:
S_OK on success or COM error code
NV_NOT_A_NOVAPDF_PRINTER - invalid OEMID
NV_INVALID_PRINTER_NAME - a printer with the specified name cannot be added

Remarks:
This method must be called before printing documents and setting novaPDF options. It will add a printer in the system with the specified name. This printer will be available until DeleteNovaPrinter is called with the same printer name. Use this method if you wish to work with temporary printers.

1.4.11.4.8 AddNovaPrinter2

The AddNovaPrinter2 method adds a temporary novaPDF printer in the system

```cpp
HRESULT AddNovaPrinter2(
    [in] BSTR p_wsPrinterName,
    [in, string] BSTR p_wsOEMID,
    [in, string] BSTR p_wsServicePort,
    [in, string] BSTR p_wsLicenseKey);
```

Parameters:
p_wsPrinterName
[in] pointer to a null terminated Unicode string containing the name of the printer
p_wsOEMID
[in] custom OEMID; for trial is "nPdfSdk10_Softland"
p_wsServicePort
[in] novaPDF Server service port number
p_wsLicenseKey
[in] license key

Return values:
S_OK on success or COM error code
NV_NOT_A_NOVAPDF_PRINTER - invalid OEMID
NV_INVALID_PRINTER_NAME - a printer with the specified name cannot be added

Remarks:
This method must be called before printing documents and setting novaPDF options. It will add a printer in the system with the specified name. This printer will be available until DeleteNovaPrinter is called with the same printer name. Use this method if you wish to work with temporary printers.

1.4.11.4.9 AddNovaPrinterExt

The AddNovaPrinterExt method adds a temporary novaPDF printer in the system

```cpp
HRESULT AddNovaPrinterExt(
    [in, string] LPCWSTR p_wsPrinterName,
    [in, string] LPCWSTR p_wsPortName,
    [in, string] LPCWSTR p_wsOEMID,
    [in, string] LPCWSTR p_wsServicePort,
    [in, string] LPCWSTR p_wsLicenseKey,
```
[in] SHORT p_nDefaultPaperSize,
[in, string] LPCWSTR p_wsDefaultPaperName,
[in] SHORT p_nDefaultPaperLength,
[in] SHORT p_nDefaultPaperWidth,
[in] SHORT p_nDefaultResolution,
[in] SHORT p_nMaxCopies,
[in] SHORT p_nDefaultCopies,
[in] SHORT p_nDefaultCollate,
[in] SHORT p_nDefaultOrientation,
[in] SHORT p_nDefaultScale,
[in] BOOL p_bAllowChangePaper,
[in] BOOL p_bAllowChangeResolution,
[in] BOOL p_bAllowChangeCopies,
[in] BOOL p_bAllowChangeCollate,
[in] BOOL p_bAllowChangeOrientation,
[in] BOOL p_bAllowChangeScale,
[in] BOOL p_bSharedPrinter,
[in, string] LPCWSTR p_wsSharedName
);

Parameters:

- **p_wsPrinterName**: [in] pointer to a null terminated Unicode string containing the name of the printer
- **p_wsPortName**: [in] pointer to a null terminated Unicode string containing the name of the printer port
- **p_wsOEMID**: [in] pointer to a null terminated Unicode string containing custom OEMID; for trial is "nPdfSdk10_Softland"
- **p_wsServicePort**: [in] pointer to a null terminated Unicode string containing novaPDF Server service port number, by default 8501
- **p_wsLicenseKey**: [in] pointer to a null terminated Unicode string containing license key
- **p_nDefaultPaperSize**: [in] default paper size (for instance 1 for Letter, 9 for A4,..)
- **p_wsDefaultPaperName**: [in] pointer to a null terminated Unicode string containing paper name (like Letter, A4,...)
- **p_nDefaultPaperLength**: [in] default paper length expressed in tenths of millimeters (for A4, 2970)
- **p_nDefaultPaperWidth**: [in] default paper width expressed in tenths of millimeters (for A4, 2100)
- **p_nDefaultResolution**: [in] default resolution (300, 600, ...)
- **p_nMaxCopies**: [in] maximum number of copies allowed (999)
- **p_nDefaultCopies**: [in] default number of copies (1)
- **p_nDefaultCollate**: [in] default collate (1 or 0)
- **p_nDefaultOrientation**: [in] default orientation (1 - portrait, 2 - landscape)
- **p_nDefaultScale**: [in] default scale (100)
- **p_bAllowChangePaper**: [in] bool, allow user to change paper size
- **p_bAllowChangeResolution**: [in] bool, allow user to change resolution
p_bAllowChangeCopies
  [in] bool, allow user to change copies

p_bAllowChangeCollate
  [in] bool, allow user to change collate

p_bAllowChangeOrientation
  [in] bool, allow user to change orientation

p_bAllowChangeScale
  [in] bool, allow user to change scale

p_bSharedPrinter
  [in] share the printer when added

p_wsSharedName
  [in] pointer to a null terminated Unicode string containing shared printer name

Return values:
S_OK on success or COM error code
NV_NOT_A_NOVAPDF_PRINTER - invalid OEMID
NV_INVALID_LICENSE - invalid license
NV_INVALID_PRINTER_NAME - a printer with the specified name or options cannot be added

Remarks:
This method must be called before printing documents and setting novaPDF options. It will add a printer in the system with the specified name and options. This printer will be available until DeleteNovaPrinter is called with the same printer name. Use this method if you wish to work with temporary printers and you wish to configure special printer defaults.

1.4.11.4.10 AddNovaPrinterExt2

The AddNovaPrinterExt2 method adds a temporary novaPDF printer in the system

HRESULT AddNovaPrinterExt2(
  [in,  string] BSTR p_wsPrinterName,
  [in,  string] BSTR p_wsPortName,
  [in,  string] BSTR p_wsOEMID,
  [in,  string] BSTR p_wsServicePort,
  [in,  string] BSTR p_wsLicensekey,
  [in]  SHORT p_nDefaultPaperSize,
  [in,  string] BSTR p_wsDefaultPaperName,
  [in]  SHORT p_nDefaultPaperLength,
  [in]  SHORT p_nDefaultPaperWidth,
  [in]  SHORT p_nDefaultResolution,
  [in]  SHORT p_nMaxCopies,
  [in]  SHORT p_nDefaultCopies,
  [in]  SHORT p_nDefaultCollate,
  [in]  SHORT p_nDefaultOrientation,
  [in]  SHORT p_nDefaultScale,
  [in]  BOOL p_bAllowChangePaper,
  [in]  BOOL p_bAllowChangeResolution,
  [in]  BOOL p_bAllowChangeCopies,
  [in]  BOOL p_bAllowChangeCollate,
  [in]  BOOL p_bAllowChangeOrientation,
  [in]  BOOL p_bAllowChangeScale,
  [in]  BOOL p_bSharedPrinter,
  [in,  string] BSTR p_wsSharedName
Parameters:

- p_wsPrinterName
  - [in] printer name
- p_wsPortName
  - [in] printer port
- p_wsOEMID
  - [in] custom OEMID; for trial is "nPdfSdk10_Softland"
- p_wsServicePort
  - [in] novaPDF Server service port number, by default 8501
- p_wsLicenseKey
  - [in] license key
- p_nDefaultPaperSize
  - [in] default paper size (for instance 1 for Letter, 9 for A4, ..)
- p_wsDefaultPaperName
  - [in] pointer to a null terminated Unicode string containing paper name (like Letter)
- p_nDefaultPaperLength
  - [in] default paper length expressed in tenths of millimeters (for A4, 2970)
- p_nDefaultPaperWidth
  - [in] default paper width expressed in tenths of millimeters (for A4, 2100)
- p_nDefaultResolution
  - [in] default resolution (300, 600, ..)
- p_nMaxCopies
  - [in] maximum number of copies allowed (999)
- p_nDefaultCopies
  - [in] default number of copies (1)
- p_nDefaultCollate
  - [in] default collate (1 or 0)
- p_nDefaultOrientation
  - [in] default orientation (1 - portrait, 2 - landscape)
- p_nDefaultScale
  - [in] default scale (100)
- p_bAllowChangePaper
  - [in] bool, allow user to change paper size
- p_bAllowChangeResolution
  - [in] bool, allow user to change resolution
- p_bAllowChangeCopies
  - [in] bool, allow user to change copies
- p_bAllowChangeCollate
  - [in] bool, allow user to change collate
- p_bAllowChangeOrientation
  - [in] bool, allow user to change orientation
- p_bAllowChangeScale
  - [in] bool, allow user to change scale
- p_bSharedPrinter
  - [in] share the printer when added
- p_wsSharedName
  - [in] shared printer name

Return values:

- S_OK on success or COM error code
- NV_NOT_A_NOVAPDF_PRINTER - invalid OEMID
- NV_INVALID_LICENSE - invalid license
- NV_INVALID_PRINTER_NAME - a printer with the specified name or options cannot be added
Remarks:
This method must be called before printing documents. It will add a printer in the system with the specified name and options. This printer will be available until DeleteNovaPrinter2 is called with the same printer name. Use this method if you wish to work with temporary printers and you wish to configure special printer defaults.

1.4.11.4.11 AddOverlay

The AddOverlay method adds an overlay

```c
HRESULT AddOverlay(
    [out, string] LPWSTR* p_pwsNewOverlayId,
    [out, string] LPWSTR* p_pwsNewLayoutId,
);
```

Parameters:
- `p_pwsNewOverlayId`
  - [out] new overlay id
- `p_pwsNewLayoutId`
  - [out] new layout id

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded

Remarks:
This method adds a new action at the end of the list of actions. After adding an action, you can change action position in the list and set specific options for each action type.

1.4.11.4.12 AddOverlay2

The AddOverlay2 method adds an overlay

```c
HRESULT AddOverlay2(
    [out, string] BSTR* p_pwsNewOverlayId,
    [out, string] BSTR* p_pwsNewLayoutId,
);
```

Parameters:
- `p_pwsNewOverlayId`
  - [out] new overlay id
- `p_pwsNewLayoutId`
  - [out] new layout id

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded

Remarks:
This method adds a new action at the end of the list of actions. After adding an action, you can change action position in the list and set specific options for each action type.

1.4.11.4.13 AddProfile

The AddProfile method adds a new profile

```c
HRESULT AddProfile(
    [in, string] LPCWSTR p_wsProfileName,
    [in] BOOL    p_bPublicProfile
    [out, string] LPWSTR* p_pwsNewProfileId
);
```

**Parameters:**
- `p_wsProfileName` [in] name of the profile to add
- `p_bPublicProfile` [in] Flag if the profile is a public or a private profile
- `p_pwsNewProfileId` [out] return profile id

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_SERVICE_ERROR - cannot connect to novaPDF Server service
- NV_PROFILE_ERROR - cannot read default profile
- NV_PROFILE_SAVE_ERROR - cannot save new profile

**Remarks:**
The newly created profile contains default settings.

1.4.11.4.14 AddProfile2

The AddProfile2 method adds a new profile

```c
HRESULT AddProfile2(
    [in, string] BSTR p_wsProfileName,
    [in] BOOL    p_bPublicProfile
    [out, string] BSTR* p_pwsNewProfileId
);
```

**Parameters:**
- `p_wsProfileName` [in] name of the profile to add
- `p_bPublicProfile` [in] Flag if the profile is a public or a private profile
- `p_pwsNewProfileId` [out] return profile id

**Return values:**
- S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_SERVICE_ERROR - cannot connect to novaPDF Server service
NV_PROFILE_ERROR - cannot read default profile
NV_PROFILE_SAVE_ERROR - cannot save new profile

Remarks:
The newly created profile contains default settings.

1.4.11.4.15 AddSaveRule

The AddSaveRule method adds a save rule.

```c
HRESULT AddSaveRule(
    [in] LONG p_nType,
    [in, string] LPCWSTR p_wsName,
    [in, string] LPCWSTR p_wsDescription,
    [in, string] LPCWSTR p_wsText,
    [in, string] LPCWSTR p_wsFormat,
    [in] BOOL p_bEnabled,
    [out] LONG* p_pnIndex
);
```

Parameters:
- `p_nType` [in] save rule type (0 - remove from start, 1 - remove from end, 2 - remove all, 3 - regular expression)
- `p_wsName` [in] rule name
- `p_wsDescription` [in] rule description
- `p_wsText` [in] text to remove or regular expression
- `p_wsFormat` [in] format for regular expression
- `p_bEnabled` [in] flag, rule is enabled
- `p_pnIndex` [out] rule index

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded

Remarks:
This method adds a new save rule at the end of the list.

1.4.11.4.16 AddSaveRule2

The AddSaveRule2 method adds a save rule.

```c
HRESULT AddSaveRule2(
    [in] LONG p_nType,
    [in, string] LPCWSTR p_wsName,
    [in, string] LPCWSTR p_wsDescription,
    [in, string] LPCWSTR p_wsText,
    [in, string] LPCWSTR p_wsFormat,
    [in] BOOL p_bEnabled,
    [out] LONG* p_pnIndex
);
```
Parameters:

- **p_nType**: [in] save rule type (0-remove from start, 1- remove from end, 2 - remove all, 3 - regular expression)
- **p_wsName**: [in] rule name
- **p_wsDescription**: [in] rule description
- **p_wsText**: [in] text to remove or regular expression
- **p_wsFormat**: [in] format for regular expression
- **p_bEnabled**: [in] flag, rule is enabled
- **p_pnIndex**: [out] rule index

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded

Remarks:

This method adds a new save rule at the end of the list.

### 1.4.11.4.17 AddUserTag

The **AddUserTag** method adds an user tag

```c
HRESULT AddUserTag(
    [in, string] LPCWSTR p_wsTag,
    [in, string] LPCWSTR p_wsDefaultValue,
    [in] BOOL p_bEnabled,
    [out] LONG* p_pnIndex
);
```

Parameters:

- **p_wsTag**: [in] user tag
- **p_wsDefaultValue**: [in] default tag value
- **p_bEnabled**: [in] flag, tag is enabled
- **p_pnIndex**: [out] tag index
Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded

Remarks:
This method adds a new user tag at the end of the list.

### 1.4.11.4.18 AddUserTag2

The AddUserTag2 method adds an user tag

```c
HRESULT AddUserTag2(
    [in, string] BSTR p_wsTag, // user tag
    [in, string] BSTR p_wsDefaultValue, // default tag value
    [in] BOOL p_bEnabled, // flag, tag is enabled
    [out] LONG* p_pnIndex // tag index
);
```

Parameters:
- `p_wsTag` [in] user tag
- `p_wsDefaultValue` [in] default tag value
- `p_bEnabled` [in] flag, tag is enabled
- `p_pnIndex` [out] tag index

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded

Remarks:
This method adds a new user tag at the end of the list.

### 1.4.11.4.19 AddWatermarkImage

The AddWatermarkImage method adds a new image watermark.

```c
HRESULT AddWatermarkImage(
    [out, string] LPWSTR* p_pwsNewWatermarkId, // return new watermark id
    [out, string] LPWSTR* p_pwsNewLayoutId
);
```

Parameters:
- `p_pwsNewWatermarkId` [out] - return new watermark id
- `p_pwsNewLayoutId`
AddWatermarkImage2

The AddWatermarkImage2 method adds a new image watermark.

```c
HRESULT AddWatermarkImage2(
    [out, string] BSTR* p_pwsNewWatermarkId,
    [out, string] BSTR* p_pwsNewLayoutId
);
```

**Parameters:**
- `p_pwsNewWatermarkId`
  - [out, string] - return new watermark id
- `p_pwsNewLayoutId`
  - [in, string] - return layout id for the new watermark

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded

**Remarks:**
The newly created watermark contains default settings. Use the watermark id and layout id as parameters for set watermark and layout options functions. For layout options, see Working with Layout objects.

AddWatermarkText

The AddWatermarkText method adds a new text watermark.

```c
HRESULT AddWatermarkText(
    [out, string] LPWSTR* p_pwsNewWatermarkId,
    [out, string] LPWSTR* p_pwsNewLayoutId
);
```

**Parameters:**
- `p_pwsNewWatermarkId`
  - [out, string] - return new watermark id
- `p_pwsNewLayoutId`
  - [in, string] - return layout id for the new watermark

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded

**Remarks:**
The newly created watermark contains default settings. Use the watermark id and layout id as parameters for set watermark and layout options functions. For layout options, see Working with Layout objects.
p_pwsNewLayoutId
  [in, string] - return layout id for the new watermark

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded

Remarks:
The newly created watermark contains default settings. Use the watermark id and layout id as
parameters for set watermark and layout options functions.
For layout options, see Working with Layout objects.

1.4.11.4.22 AddWatermarkText2

The AddWatermarkText2 method adds a new text watermark.

HRESULT AddWatermarkText2(
  [out, string] BSTR* p_pwsNewWatermarkId,
  [out, string] BSTR* p_pwsNewLayoutId
);

Parameters:
p_pwsNewWatermarkId,
  [out, string] - return new watermark id
p_pwsNewLayoutId
  [in, string] - return layout id for the new watermark

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded

Remarks:
The newly created watermark contains default settings. Use the watermark id and layout id as
parameters for set watermark and layout options functions.
For layout options, see Working with Layout objects.

1.4.11.4.23 ClearCustomProperties

The ClearCustomProperties method deletes all custom properties

HRESULT ClearCustomProperties();

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
1.4.11.4.24 ClearFontOptions

The ClearFontOptions empties the always embed and never embed fonts lists.

```c
HRESULT ClearFontOptions(void);
```

**Parameters:**

none

**Return values:**

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_NO_PROFILE - no profile loaded

1.4.11.4.25 ClearSaveRules

The ClearSaveRules method deletes all save rules

```c
HRESULT ClearSaveRules();
```

**Return values:**

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded

1.4.11.4.26 ClearUserTags

The ClearUserTags method deletes all user tags

```c
HRESULT ClearUserTags();
```

**Return values:**

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded

1.4.11.4.27 ConvertExcelDocument

The ConvertExcelDocument converts an Excel document to PDF. Excel sheet names can be added as bookmarks in the PDF.

```c
HRESULT ConvertExcelDocument(
    [in, string] LPCWSTR p_wsDocName,
    [in] BOOL p_bConvertSheetNamesToPDFBookmarks,
    [in] BOOL p_bWaitPDF
);
```

**Parameters:**

p_wsDocName
[in] Excel document name with full path
p_bConvertSheetNamesToPDFBookmarks
[in] flag, if set to 1 Excel sheet names will be added as PDF bookmarks
p_bWaitPDF
  [in] flag, if set function returns after pdf is saved and all actions executed

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_OPENFILE_ERROR - failed to open the document
  NV_FILEPROCESS_ERROR - failed to read document information

Remarks:
Hidden links from Excel document are not converted to PDF links.

1.4.11.4.28 ConvertExcelDocument2

The ConvertExcelDocument2 converts an Excel document to PDF. Excel sheet names can be added as bookmarks in the PDF.

HRESULT ConvertExcelDocument2(
    [in, string] BSTR p_wsDocName,
    [in] BOOL p_bConvertSheetNamesToPDFBookmarks,
    [in] BOOL p_bWaitPDF
);

Parameters:
  p_wsDocName
    [in] Excel document name with full path
  p_bConvertSheetNamesToPDFBookmarks
    [in] flag, if set to 1 Excel sheet names will be added as PDF bookmarks
  p_bWaitPDF
    [in] flag, if set function returns after pdf is saved and all actions executed

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_OPENFILE_ERROR - failed to open the document
  NV_FILEPROCESS_ERROR - failed to read document information

Remarks:
Hidden links from Excel document are not converted to PDF links.

1.4.11.4.29 ConvertPowerPointDocument

The ConvertPowerPointDocument converts a PowerPoint document to PDF. There can be added a PDF bookmark for each PowerPoint slide and Internet and internal document links can be converted to PDF hyperlinks.

HRESULT ConvertPowerPointDocument(
    [in, string] LPCWSTR p_wsDocName,
    [in] BOOL p_bAddPDFBookmarkForEachSlide,
    [in] BOOL p_bConvertCrossReferenceLinks,
    [in] BOOL p_bConvertCrossDocumentLinks,
    [in] BOOL p_bConvertInternetLinks,
    [in] BOOL p_bUseRelativePaths,
    [in] BOOL p_bChangeFileExtensionToPDF,
    [in] BOOL p_bOpenFileInNewTab,
[in] BOOL p_bWaitPDF
);

Parameters:

p_wsDocName
[in] PowerPoint document name with full path

p_bAddPDFBookmarkForEachSlide
[in] flag, if set to 1 PowerPoint slides names will be added as PDF bookmarks

p_bConvertCrossReferenceLinks
[in] flag, if set to 1 cross reference links (links inside the document) will be converted to PDF hyperlinks

p_bConvertCrossDocumentLinks
[in] flag, if set to 1 cross document links (links to other documents) will be converted to PDF hyperlinks

p_bConvertInternetLinks
[in] flag, if set to 1 Internet links will be converted to PDF hyperlinks

p_bUseRelativePaths
[in] flag, for cross document links, use relative paths for files

p_bChangeFileExtensionToPDF
[in] flag, for cross document links, change file extension to .pdf

p_bOpenFileInNewTab
[in] flag, for pdf documents links, open them in a new tab in Adobe Reader

p_bWaitPDF
[in] flag, if set function returns after pdf is saved and all actions executed

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_OPENFILE_ERROR - failed to open the document
NV_FILEPROCESS_ERROR - failed to read document information

1.4.11.4.30 ConvertPowerPointDocument2

The ConvertPowerPointDocument2 converts a PowerPoint document to PDF. There can be added a PDF bookmark for each PowerPoint slide and Internet and internal document links can be converted to PDF hyperlinks.

HRESULT ConvertPowerPointDocument2(
    [in, string] BSTR p_wsDocName,
    [in] BOOL p_bAddPDFBookmarkForEachSlide,
    [in] BOOL p_bConvertCrossReferenceLinks,
    [in] BOOL p_bConvertCrossDocumentLinks,
    [in] BOOL p_bConvertInternetLinks,
    [in] BOOL p_bUseRelativePaths,
    [in] BOOL p_bChangeFileExtensionToPDF,
    [in] BOOL p_bOpenFileInNewTab,
    [in] BOOL p_bWaitPDF
);

Parameters:

p_wsDocName
[in] PowerPoint document name with full path

p_bAddPDFBookmarkForEachSlide
[in] flag, if set to 1 PowerPoint slides names will be added as PDF bookmarks
The ConvertPublisherDocument converts a Publisher document to PDF. Internet and internal document links can be converted to PDF hyperlinks.

```csharp
HRESULT ConvertPublisherDocument(
    [in, string] LPCWSTR p_wsDocName,
    [in] BOOL p_bConvertCrossReferenceLinks,
    [in] BOOL p_bConvertCrossDocumentLinks,
    [in] BOOL p_bConvertInternetLinks,
    [in] BOOL p_bConvertLinksInHeaderFooter,
    [in] BOOL p_bUseRelativePaths,
    [in] BOOL p_bChangeFileExtensionToPDF,
    [in] BOOL p_bOpenFileInNewTab,
    [in] BOOL p_bWaitPDF
);
```

Parameters:
- **p_wsDocName**
  - [in] Publisher document name with full path
- **p_bConvertCrossReferenceLinks**
  - [in] flag, if set to 1 cross reference links (links inside the document) will be converted to PDF hyperlinks
- **p_bConvertCrossDocumentLinks**
  - [in] flag, if set to 1 cross document links (links to other documents) will be converted to PDF hyperlinks
- **p_bConvertInternetLinks**
  - [in] flag, if set to 1 Internet links will be converted to PDF hyperlinks
- **p_bConvertLinksInHeaderFooter**
  - [in] flag, if set to 1 links from header and footer will be converted to PDF hyperlinks
- **p_bUseRelativePaths**
  - [in] flag, for cross document links, use relative paths for files
- **p_bChangeFileExtensionToPDF**
  - [in] flag, for cross document links, change file extension to .pdf
- **p_bOpenFileInNewTab**
  - [in] flag, for pdf documents links, open them in a new tab in Adobe Reader
- **p_bWaitPDF**
  - [in] flag, if set function returns after pdf is saved and all actions executed

Return values:
- **S_OK** on success or COM error code
- **NV_NOT_INITIALIZED** - Initialize was not called
- **NV_OPENFILE_ERROR** - failed to open the document
- **NV_FILEPROCESS_ERROR** - failed to read document information
p_bOpenFileInNewTab
    [in] flag, for pdf documents links, open them in a new tab in Adobe Reader
p_bWaitPDF
    [in] flag, if set function returns after pdf is saved and all actions executed

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_OPENFILE_ERROR - failed to open the document
NV_FILEPROCESS_ERROR - failed to read document information

1.4.11.432 ConvertPublisherDocument2

The ConvertPublisherDocument2 converts a Publisher document to PDF. Internet and internal
document links can be converted to PDF hyperlinks.

HRESULT ConvertPublisherDocument2(
    [in, string] BSTR p_wsDocName,
    [in] BOOL p_bConvertCrossReferenceLinks,
    [in] BOOL p_bConvertCrossDocumentLinks,
    [in] BOOL p_bConvertInternetLinks,
    [in] BOOL p_bConvertLinksInHeaderFooter,
    [in] BOOL p_bUseRelativePaths,
    [in] BOOL p_bChangeFileExtensionToPDF,
    [in] BOOL p_bOpenFileInNewTab,
    [in] BOOL p_bWaitPDF
);

Parameters:

p_wsDocName
    [in] Publisher document name with full path
p_bConvertCrossReferenceLinks
    [in] flag, if set to 1 cross reference links (links inside the document) will be
p_bConvertCrossDocumentLinks
    [in] flag, if set to 1 cross document links (links to other documents) will be
p_bConvertInternetLinks
    [in] flag, if set to 1 Internet links will be converted to PDF hyperlinks
p_bConvertLinksInHeaderFooter
    [in] flag, if set to 1 links from header and footer will be converted to PDF hyperlinks
p_bUseRelativePaths
    [in] flag, for cross document links, use relative paths for files
p_bChangeFileExtensionToPDF
    [in] flag, for cross document links, change file extension to .pdf
p_bOpenFileInNewTab
    [in] flag, for pdf documents links, open them in a new tab in Adobe Reader
p_bWaitPDF
    [in] flag, if set function returns after pdf is saved and all actions executed

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_OPENFILE_ERROR - failed to open the document
NV_FILEPROCESS_ERROR - failed to read document information
1.4.11.4.33  ConvertVisioDocument

The ConvertVisioDocument converts a Visio document to PDF. Internet and internal document links can be converted to PDF hyperlinks.

```
HRESULT ConvertVisioDocument(
    [in, string] LPCWSTR p_wsDocName, 
    [in] BOOL p_bConvertCrossReferenceLinks,
    [in] BOOL p_bConvertCrossDocumentLinks,
    [in] BOOL p_bConvertInternetLinks,
    [in] BOOL p_bUseRelativePaths,
    [in] BOOL p_bChangeFileExtensionToPDF,
    [in] BOOL p_bOpenFileInNewTab,
    [in] BOOL p_bWaitPDF
);
```

**Parameters:**

- **p_wsDocName**
  - [in] Visio document name with full path
- **p_bConvertCrossReferenceLinks**
  - [in] flag, if set to 1 cross reference links (links inside the document) will be converted to PDF hyperlinks
- **p_bConvertCrossDocumentLinks**
  - [in] flag, if set to 1 cross document links (links to other documents) will be converted to PDF hyperlinks
- **p_bConvertInternetLinks**
  - [in] flag, if set to 1 Internet links will be converted to PDF hyperlinks
- **p_bUseRelativePaths**
  - [in] flag, for cross document links, use relative paths for files
- **p_bChangeFileExtensionToPDF**
  - [in] flag, for cross document links, change file extension to .pdf
- **p_bOpenFileInNewTab**
  - [in] flag, for pdf documents links, open them in a new tab in Adobe Reader
- **p_bWaitPDF**
  - [in] flag, if set function returns after pdf is saved and all actions executed

**Return values:**

- S_OK on success or COM error code
- NV_NOT_INITIALIZED – Initialize was not called
- NV_OPENFILE_ERROR – failed to open the document
- NV_FILEPROCESS_ERROR – failed to read document information

1.4.11.4.34  ConvertVisioDocument2

The ConvertVisioDocument2 converts a Visio document to PDF. Internet and internal document links can be converted to PDF hyperlinks.

```
HRESULT ConvertVisioDocument2(
    [in, string] BSTR p_wsDocName,
    [in] BOOL p_bConvertCrossReferenceLinks,
    [in] BOOL p_bConvertCrossDocumentLinks,
```
Parameters:

p_wsDocName
[in] Visio document name with full path

p_bConvertCrossReferenceLinks
[in] flag, if set to 1 cross reference links (links inside the document) will be converted to PDF hyperlinks

p_bConvertCrossDocumentLinks
[in] flag, if set to 1 cross document links (links to other documents) will be converted to PDF hyperlinks

p_bConvertInternetLinks
[in] flag, if set to 1 Internet links will be converted to PDF hyperlinks

p_bUseRelativePaths
[in] flag, for cross document links, use relative paths for files

p_bChangeFileExtensionToPDF
[in] flag, for cross document links, change file extension to .pdf

p_bOpenFileInNewTab
[in] flag, for pdf documents links, open them in a new tab in Adobe Reader

p_bWaitPDF
[in] flag, if set function returns after pdf is saved and all actions executed

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_OPENFILE_ERROR - failed to open the document
NV_FILEPROCESS_ERROR - failed to read document information

1.4.11.4.35 ConvertWordDocument

The ConvertWordDocument converts a Word document to PDF. Internet and internal document links can be converted to PDF hyperlinks. Word Headings and WOrd bookmarks can be converted to PDF bookmarks.

HRESULT ConvertWordDocument(
    [in, string] LPCWSTR p_wsDocName,
    [in] BOOL p_bConvertHeadings,
    [in] BOOL p_bConvertBookmarks,
    [in] BOOL p_bConvertCrossReferenceLinks,
    [in] BOOL p_bConvertCrossDocumentLinks,
    [in] BOOL p_bConvertInternetLinks,
    [in] BOOL p_bConvertLinksInHeaderFooter,
    [in] BOOL p_bUseRelativePaths,
    [in] BOOL p_bChangeFileExtensionToPDF,
    [in] BOOL p_bOpenFileInNewTab,
    [in] BOOL p_bConvertFootEndNotesLinks,
    [in] BOOL p_bPrintComments,
    [in] UINT p_nStyleCount,
    [in, string] LPCWSTR p_pwsStyleView,
    [in] BOOL p_bConvertFootEndNotesLinks,
Parameters:

- `p_wsDocName` [in] Word document name with full path
- `p_bConvertHeadings` [in] flag, if set to 1 Word Headings will be added as PDF bookmarks
- `p_bConvertBookmarks` [in] flag, if set to 1 Word bookmarks will be added as PDF bookmarks
- `p_bConvertCrossReferenceLinks` [in] flag, if set to 1 cross reference links (links inside the document) will be converted to PDF hyperlinks
- `p_bConvertCrossDocumentLinks` [in] flag, if set to 1 cross document links (links to other documents) will be converted to PDF hyperlinks
- `p_bConvertInternetLinks` [in] flag, if set to 1 Internet links will be converted to PDF hyperlinks
- `p_bConvertLinksInHeaderFooter` [in] flag, if set to 1 links from header and footer will be converted to PDF hyperlinks
- `p_bUseRelativePaths` [in] flag, for cross document links, use relative paths for files
- `p_bChangeFileExtensionToPDF` [in] flag, for cross document links, change file extension to .pdf
- `p_bOpenFileInNewTab` [in] flag, for pdf documents links, open them in a new tab in Adobe Reader
- `p_bConvertFootEndNotesLinks` [in] flag, if set to 1 footnote and endnote links will be converted to PDF hyperlinks
- `p_bPrintComments` [in] flag, if set to 1 Word comments will be printed with the documents
- `p_nStyleCount` [in] count of styles in `p_pwsStyleView` parameter
- `p_pwsStyleView` [in] string with a list of Word headings to be converted as bookmarks; Headings are separated by `;` and for each heading there are specified heading name, heading level and heading type (for now use 0 as Standard type). Sample: "Heading 1|1|0;Heading 2|2|0"
- `p_bWaitPDF` [in] flag, if set function returns after pdf is saved and all actions executed

Return values:

- `S_OK` on success or COM error code
- `NV_NOT_INITIALIZED` - Initialize was not called
- `NV_OPENFILE_ERROR` - failed to open the document
- `NV_FILEPROCESS_ERROR` - failed to read document information

1.4.11.4.36 ConvertWordDocument2

The **ConvertWordDocument2** converts a Word document to PDF. Internet and internal document links can be converted to PDF hyperlinks. Word Headings and Word bookmarks can be converted to PDF bookmarks.

```c
HRESULT ConvertWordDocument2(
    [in, string] BSTR p_wsDocName,
    [in] BOOL p_bConvertHeadings,
    [in] BOOL p_bConvertBookmarks,
);```
[in] BOOL p_bConvertCrossReferenceLinks,
[in] BOOL p_bConvertCrossDocumentLinks,
[in] BOOL p_bConvertInternetLinks,
[in] BOOL p_bConvertLinksInHeaderFooter,
[in] BOOL p_bUseRelativePaths,
[in] BOOL p_bChangeFileExtensionToPDF,
[in] BOOL p_bOpenFileInNewTab,
[in] BOOL p_bConvertFootEndNotesLinks,
[in] BOOL p_bPrintComments,
[in] UINT p_nStyleCount,
[in, string] BSTR p_pwsStyleView,
[in] BOOL p_bWaitPDF
);

Parameters:

p_wsDocName
  [in] Word document name with full path
p_bConvertHeadings
  [in] flag, if set to 1 Word Headings will be added as PDF bookmarks
p_bConvertBookmarks
  [in] flag, if set to 1 Word bookmarks will be added as PDF bookmarks
p_bConvertCrossReferenceLinks
  [in] flag, if set to 1 cross reference links (links inside the document) will be converted to PDF hyperlinks
p_bConvertCrossDocumentLinks
  [in] flag, if set to 1 cross document links (links to other documents) will be converted to PDF hyperlinks
p_bConvertInternetLinks
  [in] flag, if set to 1 Internet links will be converted to PDF hyperlinks
p_bConvertLinksInHeaderFooter
  [in] flag, if set to 1 links from header and footer will be converted to PDF hyperlinks
p_bUseRelativePaths
  [in] flag, for cross document links, use relative paths for files
p_bChangeFileExtensionToPDF
  [in] flag, for cross document links, change file extension to .pdf
p_bOpenFileInNewTab
  [in] flag, for pdf documents links, open them in a new tab in Adobe Reader
p_bConvertFootEndNotesLinks
  [in] flag, if set to 1 footnote and endnote links will be converted to PDF hyperlinks
p_bPrintComments
  [in] flag, if set to 1 Word comments will be printed with the documents
p_nStyleCount
  [in] count of styles in p_pwsStyleView parameter
p_pwsStyleView
  [in] string with a list of Word headings to be converted as bookmarks; Headings are separated by ";" and for each heading there are specified heading name, heading level and heading type (for now use 0 as Standard type). Sample: "Heading 1;1;0;Heading 2;2;0"
p_bWaitPDF
  [in] flag, if set function returns after pdf is saved and all actions executed

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_OPENFILE_ERROR - failed to open the document
NV_FILEPROCESS_ERROR - failed to read document information
1.4.11.4.37 CopyProfile

The CopyProfile method copies an existing profile to a new profile.

```c
HRESULT CopyProfile(
    [in, string] LPCWSTR p_wsNewProfileName,
    [in] BOOL p_bPublicProfile,
    [in, string] LPCWSTR p_wsOldProfileId,
    [out] LPWSTR* p_pwsNewProfileId
);
```

Parameters:
- `p_wsNewProfileName` [in] name of the new profile
- `p_bPublicProfile` [in] Flag if the profile is a public or a private profile
- `p_wsOldProfileId` [in] id for the profile to be copied
- `p_pwsNewProfileId` [out] the new profile id

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_SERVICE_ERROR - cannot connect to novaPDF Server service
- NV_PROFILE_ERROR - cannot read default profile
- NV_PROFILE_SAVE_ERROR - cannot save new profile

1.4.11.4.38 CopyProfile2

The CopyProfile2 method copies an existing profile to a new profile.

```c
HRESULT CopyProfile2(
    [in, string] BSTR p_wsNewProfileName,
    [in] BOOL p_bPublicProfile,
    [in, string] BSTR p_wsOldProfileId,
    [out] LPWSTR* p_pwsNewProfileId
);
```

Parameters:
- `p_wsNewProfileName` [in] name of the new profile
- `p_bPublicProfile` [in] Flag if the profile is a public or a private profile
- `p_wsOldProfileId` [in] id for the profile to be copied
- `p_pwsNewProfileId` [out] the new profile id

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_SERVICE_ERROR - cannot connect to novaPDF Server service
- NV_PROFILE_ERROR - cannot read default profile
- NV_PROFILE_SAVE_ERROR - cannot save new profile
1.4.11.4.39 DeleteAction

The DeleteAction method deletes an existing action.

```c
HRESULT DeleteAction(
    [in, string] LPCWSTR p_wsActionId
);
```

Parameters:
- `p_wsActionId` [in] action id

Return values:
- `S_OK` on success or COM error code
- `NV_NOT_INITIALIZED` - Initialize was not called
- `NV_UNKNOWN_PROFILE` - no profile loaded
- `NV_INVALID_OPTION` - id not a valid guid
- `NV_INVALID_ACTION` - invalid action id

1.4.11.4.40 DeleteAction2

The DeleteAction2 method deletes an existing action.

```c
HRESULT DeleteAction(
    [in, string] BSTR p_wsActionId
);
```

Parameters:
- `p_wsActionId` [in] action id

Return values:
- `S_OK` on success or COM error code
- `NV_NOT_INITIALIZED` - Initialize was not called
- `NV_UNKNOWN_PROFILE` - no profile loaded
- `NV_INVALID_OPTION` - id not a valid guid
- `NV_INVALID_ACTION` - invalid action id

1.4.11.4.41 DeleteBookmarkDefinition

The DeleteBookmarkDefinition method deletes an existing bookmark definition.

```c
HRESULT DeleteBookmarkDefinition(
    [in] SHORT p_nDefinition
);
```

Parameters:
- `p_nDefinition` [in] definition index
Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_INVALID_BOOKMARK_DEF - wrong bookmark definition index
NV_UNKNOWN_PROFILE - no profile loaded

1.4.11.4.42 DeleteCustomProperty

The **DeleteCustomProperty** method deletes the custom property for the given index

```c
HRESULT DeleteCustomProperty(
    [in] LONG p_nIndex
);
```

**Parameters:**
- `p_nIndex` [in] property index

**Return values:**
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_INVALID_CUSTOM_PROPERTY - invalid custom property index

**Remarks:**
This method deletes the custom property at position given by `p_nIndex`

1.4.11.4.43 DeleteLayout

The **DeleteLayout** method deletes an existing layout for the specified object

```c
HRESULT DeleteLayout(
    [in] LPCWSTR p_wsObjectId
    [in] LPCWSTR p_wsLayoutId
);
```

**Parameters:**
- `p_wsObjectId` [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_wsLayoutId` [in] layout id (obtained with GetLayout)

**Return values:**
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_INVALID_LAYOUT - layout not found for specified object

1.4.11.4.44 DeleteLayout2

The **DeleteLayout2** method deletes an existing layout for the specified object

```c
HRESULT DeleteLayout2(
    [in] BSTR p_wsObjectId
);
```
[in] BSTR p_wsLayoutId

Parameters:
  p_pwsObjectId
    [in] object id (watermark text, watermark image, overlay, signature or content)
  p_pwsLayoutId
    [in] layout id (obtained with GetLayout2)

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_UNKNOWN_PROFILE - no profile loaded
  NV_INVALID_LAYOUT - layout not found for specified object

1.4.11.4.45 DeleteNovaPrinter

The **DeleteNovaPrinter** method adds a temporary novaPDF printer in the system

```cpp
HRESULT DeleteNovaPrinter(
    [in] LPCWSTR p_wsPrinterName);
```

Parameters:
  p_wsPrinterName
    [in] pointer to a null terminated Unicode string containing the name of the printer to be deleted.

Return values:
  S_OK on success or COM error code
  NV_INVALID_PRINTER_NAME - not a novaPDF printer
  NV_ERROR_DELETE_PRINTER - there was an error when deleting the printer

Remarks:
This method must be called for printers added with AddNovaPrinter method, after finish printing documents. It will delete the temporary printer from the system. Use this method if you wish to work with temporary printers.

1.4.11.4.46 DeleteNovaPrinter2

The **DeleteNovaPrinter2** method adds a temporary novaPDF printer in the system

```cpp
HRESULT DeleteNovaPrinter2(
    [in] BSTR p_wsPrinterName);
```

Parameters:
  p_wsPrinterName
    [in] name of the printer to be deleted.

Return values:
  S_OK on success or COM error code
  NV_INVALID_PRINTER_NAME - not a novaPDF printer
  NV_ERROR_DELETE_PRINTER - there was an error when deleting the printer

Remarks:
This method must be called for printers added with AddNovaPrinter method, after finish printing documents. It will delete the temporary printer from the system. Use this method if you wish to work with temporary printers.

### 1.4.11.4.47 DeleteOverlay

The **DeleteOverlay** method deletes an existing overlay.

```c
HRESULT DeleteOverlay([in, string] LPCWSTR p_wsOverlayId);
```

**Parameters:**
- `p_wsOverlayId` [in] pointer to a null terminated Unicode string containing the overlay id

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OVERLAY - wrong overlay index

### 1.4.11.4.48 DeleteOverlay2

The **DeleteOverlay2** method deletes an existing overlay.

```c
HRESULT DeleteOverlay2([in, string] BSTR p_wsOverlayId);
```

**Parameters:**
- `p_wsOverlayId` [in] overlay id

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OVERLAY - wrong overlay index

### 1.4.11.4.49 DeleteOverrideOptions

The **DeleteOverrideOptions** method removes all override options

```c
HRESULT DeleteOverrideOptions();
```

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - error deleting options
1.4.11.4.50 DeletePrinterActivePublicProfile

The `DeletePrinterActivePublicProfile` method removes the active public profile setting for a printer.

```c
HRESULT tDeletePrinterActivePublicProfile(
    [in, string] LPCWSTR p_wsPrinterName
);
```

**Parameters:**
- `p_wsPrinterName` [in, string] printer name

**Return values:**
- S_OK on success or COM error code
- NVSERVICE_ERROR - error connecting to novaPDF Server service

**Remarks:**
Removes the active profile setting for the printer. The users will be allowed to change the active profile.

1.4.11.4.51 DeletePrinterActivePublicProfile2

The `DeletePrinterActivePublicProfile2` method removes the active public profile setting for a printer.

```c
HRESULT tDeletePrinterActivePublicProfile2(
    [in, string] BSTR p_wsPrinterName
);
```

**Parameters:**
- `p_wsPrinterName` [in, string] printer name

**Return values:**
- S_OK on success or COM error code
- NVSERVICE_ERROR - error connecting to novaPDF Server service

**Remarks:**
Removes the active profile setting for the printer. The users will be allowed to change the active profile.

1.4.11.4.52 DeleteProfile

The `DeleteProfile` method deletes an existing profile.

```c
HRESULT DeleteProfile(
    [in] LPCWSTR p_wsProfileId
);
```

**Parameters:**
- `p_wsProfileId` [in] pointer to a null terminated Unicode string containing the id of the profile

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NVSERVICE_ERROR - error connecting to novaPDF Server service
- NVPROFILE_DELETE_ERROR - error deleting profile
- NVPROFILE_ERROR - error reading profiles
1.4.11.4.53 DeleteProfile2

The **DeleteProfile2** method deletes an existing profile.

```c
HRESULT DeleteProfile2(
    [in] BSTR p_wsProfileId
);
```

**Parameters:**

- `p_wsProfileId` *(in)* the id of the profile to delete.

**Return values:**

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_SERVICE_ERROR - error connecting to novaPDF Server service
- NV_PROFILE_DELETE_ERROR - error deleting profile
- NV_PROFILE_ERROR - error reading profiles

1.4.11.4.54 DeleteSaveRule

The **DeleteSaveRule** method deletes the save rule for the given index.

```c
HRESULT DeleteSaveRule(
    [in] LONG p_nIndex
);
```

**Parameters:**

- `p_nIndex` *(in)* tag index

**Return values:**

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_SAVE_RULE - invalid save rule index

**Remarks:**

This method deletes the save rule at position given by p_nIndex.

1.4.11.4.55 DeleteUserTag

The **DeleteUserTag** method deletes the user tag for the given index.

```c
HRESULT DeleteUserTag(
    [in] LONG p_nIndex
);
```

**Parameters:**

- `p_nIndex` *(in)* tag index
Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_USER_TAG - invalid user tag index

Remarks:
This method deletes the tag at position given by p_nIndex

1.4.11.4.56 DeleteWatermarkImage

The **DeleteWatermarkImage** method deletes an existing watermark image.

```c
HRESULT DeleteWatermarkImage(
    [in, string] LPCWSTR p_wsWatermarkId
);
```

Parameters:
- `p_wsWatermarkId` [in] pointer to a null terminated Unicode string containing the watermark id

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_IMG - wrong image watermark index

1.4.11.4.57 DeleteWatermarkImage2

The **DeleteWatermarkImage2** method deletes an existing watermark image.

```c
HRESULT DeleteWatermarkImage2(
    [in, string] BSTR p_wsWatermarkId
);
```

Parameters:
- `p_wsWatermarkId` [in] the watermark id

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_IMG - wrong image watermark index

1.4.11.4.58 DeleteWatermarkText

The **DeleteWatermarkText** method deletes an existing watermark text.

```c
HRESULT DeleteWatermarkText(
    [in, string] LPCWSTR p_wsWatermarkId
);
```
Parameters:
  p_wsWatermarkId
  [in] pointer to a null terminated Unicode string containing the watermark id

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_UNKNOWN_PROFILE - no profile loaded
  NV_INVALID_WATERMARK_IMG - wrong text watermark index

1.4.11.459 DeleteWatermarkText2

The **DeleteWatermarkText2** method deletes an existing watermark text.

```c
HRESULT DeleteWatermarkText2([in, string] BSTR p_wsWatermarkId);```

Parameters:
  p_wsWatermarkId
  [in] the watermark id

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_UNKNOWN_PROFILE - no profile loaded
  NV_INVALID_WATERMARK_IMG - wrong text watermark index

1.4.11.460 DisableAction

The **DisableAction** method disables an existing action

```c
HRESULT DisableAction([in, string] LPCWSTR p_wsActionId);```

Parameters:
  p_wsActionId
  [in] action id

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_UNKNOWN_PROFILE - no profile loaded
  NV_INVALID_OPTION - id not a valid guid
  NV_INVALID_ACTION - invalid action id
1.4.11.4.61 DisableAction2

The DisableAction2 method disables an existing action.

```c
HRESULT DisableAction2(
    [in, string] BSTR p_wsActionId
);
```

**Parameters:**
- `p_wsActionId` [in] action id

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - id not a valid guid
- NV_INVALID_ACTION - invalid action id

1.4.11.4.62 DisableActionType

The DisableActionType method disables an existing action.

```c
HRESULT DisableActionType(
    [in] LONG p_nActionType
);
```

**Parameters:**
- `p_nActionType` [in] action type

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - id not a valid guid
- NV_INVALID_ACTION - invalid action id

1.4.11.4.63 DisableCustomProperty

The DisableCustomProperty method disables the custom property for the given index.

```c
HRESULT DisableCustomProperty(
    [in] LONG p_nIndex
);
```

**Parameters:**
- `p_nIndex` [in] tag index

**Return values:**
- S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_CUSTOM_PROPERTY - invalid custom property index

Remarks:
This method disables the custom property at position given by p_nIndex

1.4.11.4.64 DisableSaveRule

The DisableSaveRule method disables the save rule for the given index

HRESULT DisableSaveRule(
    [in] LONG p_nIndex
);

Parameters:
p_nIndex
    [in] tag index

Return values:
    S_OK on success or COM error code
    NV_NOT_INITIALIZED - Initialize was not called
    NV_UNKNOWN_PROFILE - no profile loaded
    NV_INVALID_SAVE_RULE - invalid save rule index

Remarks:
This method disables the save rule at position given by p_nIndex

1.4.11.4.65 DisableUserTag

The DisableUserTag method disables the user tag for the given index

HRESULT DisableUserTag(
    [in] LONG p_nIndex
);

Parameters:
p_nIndex
    [in] property index

Return values:
    S_OK on success or COM error code
    NV_NOT_INITIALIZED - Initialize was not called
    NV_UNKNOWN_PROFILE - no profile loaded
    NV_INVALID_USER_TAG - invalid user tag index

Remarks:
This method disables the tag at position given by p_nIndex
1.4.11.4.66  EnableAction

The EnableAction method enables an existing action

```c
HRESULT EnableAction(
    [in, string] LPCWSTR p_wsActionId
);
```

Parameters:
- `p_wsActionId`: [in] action id

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - id not a valid guid
- NV_INVALID_ACTION - invalid action id

1.4.11.4.67  EnableAction2

The EnableAction2 method enables an existing action

```c
HRESULT EnableAction2(
    [in, string] BSTR p_wsActionId
);
```

Parameters:
- `p_wsActionId`: [in] action id

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - id not a valid guid
- NV_INVALID_ACTION - invalid action id

1.4.11.4.68  EnableActionType

The EnableActionType method enables an existing action

```c
HRESULT EnableActionType(
    [in] LONG p_nActionType
);
```

Parameters:
- `p_nActionType`: [in] action type

Return values:
- S_OK on success or COM error code
1.4.11.4.69 EnableCustomProperty

The `EnableCustomProperty` method enables the custom property for the given index.

```c
HRESULT EnableCustomProperty(
    [in] LONG p_nIndex
);
```

Parameters:
- `p_nIndex` [in] property index

Return values:
- `S_OK` on success or COM error code
- `NV_NOT_INITIALIZED` - Initialize was not called
- `NV_UNKNOWN_PROFILE` - no profile loaded
- `NV_INVALID_OPTION` - id not a valid guid
- `NV_INVALID_ACTION` - invalid action id

Remarks:
This method enables the custom property at position given by `p_nIndex`

1.4.11.4.70 EnableSaveRule

The `EnableSaveRule` method enables the save rule for the given index.

```c
HRESULT EnableSaveRule(
    [in] LONG p_nIndex
);
```

Parameters:
- `p_nIndex` [in] tag index

Return values:
- `S_OK` on success or COM error code
- `NV_NOT_INITIALIZED` - Initialize was not called
- `NV_UNKNOWN_PROFILE` - no profile loaded
- `NV_INVALID_SAVE_RULE` - invalid save rule index

Remarks:
This method enables the save rule at position given by `p_nIndex`

1.4.11.4.71 EnableUserTag

The `EnableUserTag` method enables the user tag for the given index.

```c
HRESULT EnableUserTag(
    [in] LONG p_nIndex
);```
Parameters:

\[ p_{\text{nIndex}} \]

\[ \text{[in]} \text{ tag index} \]

Return values:

\[ S_{\text{OK}} \] on success or COM error code
\[ NV_{\text{NOT_INITIALIZED}} \] - Initialize was not called
\[ NV_{\text{UNKNOWN_PROFILE}} \] - no profile loaded
\[ NV_{\text{INVALID_USER_TAG}} \] - invalid user tag index

Remarks:

This method enables the tag at position given by \( p_{\text{nIndex}} \)

1.4.11.4.72 GetAction

The GetAction method retrieves the action id and action type for the specified index in the action list

\[
\text{HRESULT GetAction(}
  \text{[in]} \text{ } \text{LONG } p_{\text{nIndex}},
  \text{[out, string]} \text{ } \text{LPWSTR* } p_{\text{pwsActionId}},
  \text{[out]} \text{ } \text{LONG* } p_{\text{pntype}}
\); 

Parameters:

\[ p_{\text{nIndex}} \]

\[ \text{[in]} \text{ action index} \]
\[ p_{\text{pwsActionId}} \]

\[ \text{[out]} \text{ action id} \]
\[ p_{\text{pntype}} \]

\[ \text{[out]} \text{ action type} \]

Return values:

\[ S_{\text{OK}} \] on success or COM error code
\[ NV_{\text{NOT_INITIALIZED}} \] - Initialize was not called
\[ NV_{\text{UNKNOWN_PROFILE}} \] - no profile loaded
\[ NV_{\text{INVALID_OPTION}} \] - wrong index

1.4.11.4.73 GetAction2

The GetAction2 method retrieves the action id and action type for the specified index in the action list

\[
\text{HRESULT GetAction2(}
  \text{[in]} \text{ } \text{LONG } p_{\text{nIndex}},
  \text{[out, string]} \text{ } \text{BSTR* } p_{\text{pwsActionId}},
  \text{[out]} \text{ } \text{LONG* } p_{\text{pntype}}
\); 

Parameters:

- `p_nIndex`
  - [in] action index
- `p_pwsActionId`
  - [out] action id
- `p_pnType`
  - [out] action type

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong index

1.4.11.4.74 GetActionCount

The GetActionCount method retrieves the number of actions in the profile

```c
HRESULT GetActionCount(
    [out] LONG* p_pnCount);
```

Parameters:

- `p_pnCount`
  - [out] actions count

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded

1.4.11.4.75 GetActionOptionBool

The GetActionOptionBool method retrieves an option of boolean type for an action

```c
HRESULT GetActionOptionBool(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [out] BOOL*  p_pbValue);
```

Parameters:

- `p_pwsActionId`
  - [in] Action id
- `p_wsOption`
  - [in] option constant
- `p_pbValue`
  - [out] the value of the option
Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.76 GetActionOptionBool2

The **GetActionOptionBool2** method retrieves an option of boolean type for an action

```
HRESULT GetActionOptionBool2(
    [in, string] BSTR p_pwsActionId,
    [in]        LONG    p_nOption,
    [out]       BOOL*  p_pbValue
);
```

Parameters:
- **p_pwsActionId**
  - [in] Action id
- **p_nOption**
  - [in] option constant
- **p_pbValue**
  - [out] the value of the option

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.77 GetActionOptionEncryptedString

The **GetActionOptionEncryptedString** method retrieves an option of encrypted string type for an action
HRESULT GetActionOptionEncryptedString(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG p_nOption,
    [out, string] LPWSTR* p_pwsValue
);

Parameters:

- p_pwsActionId
  - [in] Action id
- p_wsOption
  - [in] option constant
- p_pwsValue
  - [out] the value of the option

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.478 GetActionOptionEncryptedString2

The GetActionOptionEncryptedString2 method retrieves an option of encrypted string type for an action.

HRESULT GetActionOptionEncryptedString2(
    [in, string] BSTR p_pwsActionId,
    [in] LONG p_nOption,
    [out, string] BSTR* p_pwsValue
);

Parameters:

- p_pwsActionId
  - [in] Action id
- p_wsOption
  - [in] option constant
- p_pwsValue
  - [out] the value of the option

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
1.4.11.4.79 GetActionOptionFloat

The GetActionOptionFloat method retrieves an option of float type for an action.

```c
HRESULT GetActionOptionFloat(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [out] FLOAT*  p_pfValue
);
```

**Parameters:**
- `p_pwsActionId`: [in] Action id
- `p_nOption`: [in] option constant
- `p_pfValue`: [out] the value of the option

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.80 GetActionOptionFloat2

The GetActionOptionFloat2 method retrieves an option of float type for an action.

```c
HRESULT GetActionOptionFloat2(
    [in, string] BSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [out] FLOAT*  p_pfValue
);
```

**Parameters:**
The `GetActionOptionLong` method retrieves an option of long type for an action.

```c
HRESULT GetActionOptionLong(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [out] LONG*  p_pnValue);
```

### Parameters:
- `p_pwsActionId`: [in] Action id
- `p_wsOption`: [in] option constant
- `p_pfValue`: [out] the value of the option

### Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

### Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: `novaOptions.h`, `novaOptions.pas`, `Globals.bas`. 

1.4.11.4.81 GetActionOptionLong

The `GetActionOptionLong` method retrieves an option of long type for an action.

```c
HRESULT GetActionOptionLong(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [out] LONG*  p_pnValue);
```

### Parameters:
- `p_pwsActionId`: [in] Action id
- `p_wsOption`: [in] option constant
- `p_pfValue`: [out] the value of the option

### Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

### Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: `novaOptions.h`, `novaOptions.pas`, `Globals.bas`. 

### 1.4.11.4.82 GetActionOptionLong2

The **GetActionOptionLong2** method retrieves an option of long type for an action.

```c
HRESULT GetActionOptionLong2(
    [in, string] BSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [out] LONG*  p_pnValue
);
```

**Parameters:**
- `p_pwsActionId` [in]: Action id
- `p_wsOption` [in]: Option constant
- `p_pnValue` [out]: The value of the option

**Return values:**
- `S_OK` on success or COM error code
- `NV_NOT_INITIALIZED` - Initialize was not called
- `NV_UNKNOWN_PROFILE` - No profile loaded
- `NV_INVALID_ACTION` - Wrong action id
- `NV_INVALID_OPTION` - Wrong option constant
- `NV_PROFILE_ERROR` - Cannot find option in profile
- `NV_WRONG_OPTION_TYPE` - Option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: `novaOptions.h`, `novaOptions.pas`, `Globals.bas`.

### 1.4.11.4.83 GetActionOptionString

The **GetActionOptionString** method retrieves an option of string type for an action.

```c
HRESULT GetActionOptionString(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [out, string] LPWSTR* p_pwsValue
);
```

**Parameters:**
- `p_pwsActionId` [in]: Action id
- `p_wsOption` [in]: Option constant
- `p_pwsValue` [out]: The value of the option

**Return values:**
- `S_OK` on success or COM error code
Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.84 GetActionOptionString2

The **GetActionOptionString2** method retrieves an option of string type for an action.

```c
HRESULT GetActionOptionString2(
    [in, string] BSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [out, string] BSTR* p_pwsValue
);
```

**Parameters:**
- `p_pwsActionId` [in] Action id
- `p_wsOption` [in] option constant
- `p_pwsValue` [out] the value of the option

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.85 GetActiveProfile

The **GetActiveProfile** retrieves the id of the active profile (i.e. the profile that is used for printing).

```c
HRESULT GetActiveProfile(
    [out] LPWSTR* p_wsProfileId
);
```
Parameters:

p_wsProfileId

[out] pointer to a pointer to a null terminated Unicode string that will contain

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_NO_ACTIVE_PROFILE - there is no active profile selected for the printer

1.4.11.4.86 GetActiveProfile2

The GetActiveProfile2 retrieves the id of the active profile (i.e. the profile that is used for printing).

HRESULT GetActiveProfile2(
    [out] BSTR* p_wsProfileId
);

Parameters:

p_wsProfileId

[out] pointer to a pointer to a null terminated Unicode string that will contain

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_NO_ACTIVE_PROFILE - there is no active profile selected for the printer

1.4.11.4.87 GetBookmarkDefinition

The GetBookmarkDefinition method retrieves an existing bookmark definition properties.

HRESULT GetBookmarkDefinition(
    [in] SHORT p_nDefinition,
    [out] SHORT* p_pnHeading,
    [out] BOOL* p_pbEnabled,
    [out] BOOL* p_bbDetFont,
    [out, string] LPWSTR* p_pwsDetFont,
    [out] LONG* p_pnDetFontType,
    [out] BOOL* p_bbDetStyle,
    [out] BOOL* p_bbDetBold,
    [out] BOOL* p_bbDetItalic,
    [out] BOOL* p_bbDetSize,
    [out] INT* p_pnDetSizeVal,
    [out] FLOAT* p_pnDetSizePt,
    [out] BOOL* p_bbDispColor,
    [out] LONG* p_pnDispColor,
    [out] BOOL* p_bbDispAsBold,
    [out] BOOL* p_bbDispAsItalic,
    [out] LONG* p_pnDispColor
);

Parameters:

p_nDefinition

[in] definition index

p_pnHeading
1.4.11.488 GetBookmarkDefinition2

The `GetBookmarkDefinition2` method retrieves an existing bookmark definition properties.

```csharp
HRESULT GetBookmarkDefinition2(
    [in] SHORT p_nDefinition,
    [out] SHORT* p_pnHeading,
    [out] BOOL* p_pbEnabled,
    [out] BOOL* p_pbDetFont,
    [out, string] BSTR* p_pwsDetFont,
    [out] LONG* p_pnDetFontType,
    [out] BOOL* p_pbDetStyle,
    [out] BOOL* p_pbDetBold,
    [out] BOOL* p_pbDetItalic,
    [out] BOOL* p_pbDetSize,
    [out] INT* p_pnDetSizeVal,
    [out] FLOAT* p_pnDetSizePt,
    [out] BOOL* p_pbDispAsBold,
    [out] LONG* p_pnDispColor,
    [out] FLOAT* p_pnDispSizePt,
    [out] BOOL* p_pbDetColor,
    [out] BOOL* p_pbDispAsItalic,
    [out] FLOAT* p_pnDispColor)
```

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_BOOKMARK_DEF - wrong bookmark definition index
Parameters:

- p_nDefinition [in] definition index
- p_pnHeading [out] heading index
- p_pbEnabled [out] definition is enabled
- p_pbDetFont [out] detect font flag
- p_pwsDetFont [out] font name
- p_pbDetStyle [out] detect font style
- p_pbDetBold [out] bold font
- p_pbDetItalic [out] italic font
- p_pbDetSize [out] detect font size
- p_pnDetSizeVal [out] font size
- p_pnDetSizePt [out] font size rounding
- p_pbDetColor [out] detect font color
- p_pnDetColor [out] font color (RGB value)
- p_pbDispAsBold [out] display bookmark font bold
- p_pbDispAsItalic [out] display bookmark font italic
- p_pnDispColor [out] display bookmark font color

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_BOOKMARK_DEF - wrong bookmark definition index

1.4.11.489 GetBookmarkDefinitionCount

The GetBookmarkDefinitionCount method retrieves the number of bookmark definitions in a
bookmark heading.

```c
HRESULT GetBookmarkDefinitionCount(
    [out] SHORT* p_pnCount
);
```

Parameters:

- p_pnCount [out] count of bookmark headings
1.4.11.4.90 GetContentLayout

The `GetContentLayout` method retrieves the content object and its layout object.

```c
HRESULT GetContentLayout(
    [out, string] LPWSTR* p_pwsContentId,
    [out, string] LPWSTR* p_pwsLayoutId);
```

**Parameters:**
- `p_pwsContentId`: [out, string] - content id
- `p_pwsLayoutId`: [out, string] - layout id

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded

**Remarks:**
The content layout defines how the printed content will be positioned on the PDF page. There is only one content object in the profile.

For layout options, see Working with Layout objects.

1.4.11.4.91 GetContentLayout2

The `GetContentLayout2` method retrieves the content object and its layout object.

```c
HRESULT GetContentLayout2(
    [out, string] BSTR* p_pwsContentId,
    [out, string] BSTR* p_pwsLayoutId);
```

**Parameters:**
- `p_pwsContentId`: [out, string] - content id
- `p_pwsLayoutId`: [out, string] - layout id

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded

- NV_PROFILE_ERROR - error reading profile
Remarks:
The content layout defines how the printed content will be positioned on the PDF page. There is only one content object in the profile.
For layout options, see Working with Layout objects.

1.4.11.4.92 GetCustomProperty

The GetCustomProperty method returns the custom property for the given index

```csharp
HRESULT GetCustomProperty(
    [in] LONG p_nIndex,
    [out, string] LPWSTR* p_pwsPropertyName,
    [out, string] LPWSTR* p_pwsPropertyValue,
    [out] BOOL* p_pbEnabled
);
```

Parameters:
p_nIndex
    [in] custom property index
p_pwsPropertyName
    [out] custom property name
p_pwsPropertyValue
    [out] custom property value
p_bEnabled
    [out] flag, tag is enabled

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_CUSTOM_PROPERTY - invalid custom property index

Remarks:
This method returns information about the custom property at position given by p_nIndex

1.4.11.4.93 GetCustomProperty2

The GetCustomProperty2 method returns the custom property for the given index

```csharp
HRESULT GetCustomProperty2(
    [in] LONG p_nIndex,
    [out, string] BSTR* p_pwsPropertyName,
    [out, string] BSTR* p_pwsPropertyValue,
    [out] BOOL* p_pbEnabled
);
```

Parameters:
p_nIndex

Remarks:
[in] custom property index
p_pwsPropertyName
[out] custom property name
p_pwsPropertyValue
[out] custom property value
p_bEnabled
[out] flag, tag is enabled

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_CUSTOM_PROPERTY - invalid custom property index

Remarks:
This method returns information about the custom property at position given by p_nIndex

1.4.11.4.94 GetCustomPropertiesCount

The GetCustomPropertiesCount method returns custom properties count

HRESULT GetCustomPropertiesCount(
    [out] LONG* p_pnCount
);

Parameters:
  p_pnCount
    [in] custom properties count

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded

1.4.11.4.95 GetFirstProfile

The GetFirstProfile starts an enumeration of profiles, retrieving the id of the first profile in the enumeration.

HRESULT GetFirstProfile(
    [out] LPWSTR* p_pwsProfileId
);

Parameters:
  p_pwsProfileId
    [out] pointer to a pointer to a null terminated Unicode string containing the id

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_SERVICE_ERROR - cannot connect to novaPDF Server service
NVPROFILE_ERROR - cannot load profiles
Remarks:
GetFirstProfile is used with GetNextProfile to retrieve profile names.

Sample usage:
```c
hr = GetFirstProfile(&pId);
while (SUCCEEDED(hr) && hr != NV_NO_MORE_PROFILES) {
    // do something with pName
    // ...
    CoTaskMemFree(pId);
    // get next profile if it exists
    hr = GetNextProfile(&pId);
}
```

1.4.11.4.96 GetFirstProfile2

The GetFirstProfile2 starts an enumeration of profiles, retrieving the id of the first profile in the enumeration.

```c
HRESULT GetFirstProfile2(
    [out] BSTR* p_pwsProfileId);
```

Parameters:
- p_pwsProfileId
  - [out] pointer to a pointer to a null terminated Unicode string containing the id of the first existing profile

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_SERVICE_ERROR - cannot connect to novaPDF Server service
- NV_PROFILE_ERROR - cannot load profiles

1.4.11.4.97 GetFontOption

The GetFontOption method retrieves embed options for a given font

```c
HRESULT GetFontOption(
    [in, string] LPCWSTR p_wsFontName,
    [out] BOOL* p_pbAlwaysEmbed,
    [out] BOOL* p_pbNeverEmbed);
```

Parameters:
- p_wsFontName
  - [in] font name
- p_pbAlwaysEmbed
  - [out] pointer to a boolean that will contain the always embed flag for the font
- p_pbNeverEmbed
  - [out] pointer to a boolean that will contain the never embed flag for the font

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong font name
1.4.11.4.98 GetFontOption2

The **GetFontOption2** method retrieves embed options for a given font

```c
HRESULT GetFontOption2(
    [in, string] BSTR p_wsFontName,
    [out] BOOL* p_pbAlwaysEmbed,
    [out] BOOL* p_pbNeverEmbed
);
```

**Parameters:**
- `p_wsFontName` [in] font name
- `p_pbAlwaysEmbed` [out] pointer to a boolean that will contain the always embed flag for the font
- `p_pbNeverEmbed` [out] pointer to a boolean that will contain the never embed flag for the font

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong font name

1.4.11.4.99 GetLayout

The **GetLayout** method retrieves the layout id for the specified object in the loaded profile

```c
HRESULT GetLayout(
    [in, string] LPCWSTR p_wsObjectId,
    [in] LONG p_nIndex,
    [out, string] LPWSTR* p_pwsLayoutId
);
```

**Parameters:**
- `p_wsObjectId` [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_nIndex` [in] layout index
- `p_pwsLayoutId` [out] layout id

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong object id
- NV_INVALID_LAYOUT - layout not found for specified object
Remarks:
For layout options, see Working with Layout objects.

1.4.11.4.100 GetLayout2

The GetLayout2 method retrieves the layout id for the specified object in the loaded profile.

```c
HRESULT GetLayout2(
    [in, string] BSTR p_wsObjectId,
    [in]  LONG  p_nIndex,
    [out, string] BSTR* p_pwsLayoutId
);
```

Parameters:
- `p_wsObjectId` [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_nIndex` [out] layout index
- `p_pwsLayoutId` [out] layout id

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong object id
- NV_INVALID_LAYOUT - layout not found for specified object

Remarks:
For layout options, see Working with Layout objects.

1.4.11.4.101 GetLayoutCount

The GetLayoutCount method retrieves the count of layout for the specified object in the loaded profile.

```c
HRESULT GetLayoutCount(
    [in, string] LPCWSTR p_pwsObjectId,
    [out] LONG* p_pnCount
);
```

Parameters:
- `p_pwsObjectId` [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_pnCount` [out] layouts count

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OPTION - wrong object id

1.4.11.4.102 GetLayoutCount2

The GetLayoutCount2 method retrieves the count of layout objects for the specified object in the loaded profile

HRESULT GetLayoutCount2(
    [in, string] BSTR p_pwsObjectId,
    [out] LONG* p_pnCount
);

Parameters:

p_pwsObjectId
    [in] object id (watermark text, watermark image, overlay, signature or content)

p_pnCount
    [out] layouts count

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OPTION - wrong object id

1.4.11.4.103 GetLayoutOptionBool

The GetLayoutOptionBool method retrieves an option of boolean type for a layout object

HRESULT GetLayoutOptionBool(
    [in, string] LPCWSTR p_pwsObjectId,
    [in, string] LPCWSTR p_pwsLayoutId,
    [in]  LONG  p_nOption,
    [out] BOOL* p_pbValue
);

Parameters:

p_pwsObjectId
    [in] object id (watermark text, watermark image, overlay, signature or content)

p_pwsLayoutId
    [in] layout id (obtained with GetLayout )

p_nOption
    [in] option constant

p_pbValue
    [out] will contain the value of the retrieved option

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
For layout options, see Working with Layout objects.

1.4.11.4.104 GetLayoutOptionBool2
The GetLayoutOptionBool2 method retrieves an option of string type for a layout object

```c
HRESULT GetLayoutOptionBool2(
    [in, string] BSTR p_pwsObjectId,
    [in, string] BSTR p_pwsLayoutId,
    [in]  LONG  p_nOption,
    [out] BOOL* p_pbValue
);
```

Parameters:
p_pwsObjectId
[in] object id (watermark text, watermark image, overlay, signature or content)
p_pwsLayoutId
[in] layout id (obtained with GetLayout2 )
p_nOption
[in] option constant
p_pbValue
[out] will contain the value of the retrieved option

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
For layout options, see Working with Layout objects.

1.4.11.4.105 GetLayoutOptionFloat
The GetLayoutOptionFloat method retrieves an option of float type for a layout object

```c
HRESULT GetLayoutOptionFloat(
    [in, string] LPCWSTR p_pwsObjectId,
    [in, string] LPCWSTR p_pwsLayoutId,
    [in]  LONG   p_nOption,
    [out] FLOAT* p_pfValue
);
```
The `GetLayoutOptionFloat2` method retrieves an option of float type for a layout object.

```c
HRESULT GetLayoutOptionFloat2(
    [in, string] BSTR p_pwsObjectId,
    [in, string] BSTR p_pwsLayoutId,
    [in] LONG   p_nOption,
    [out] FLOAT* p_pfValue
);
```

### Parameters:
- `p_pwsObjectId`:
  - [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_pwsLayoutId`:
  - [in] layout id (obtained with `GetLayout2`)
- `p_nOption`:
  - [in] option constant
- `p_pfValue`:
  - [out] will contain the value of the retrieved option

### Return values:
- `S_OK` on success or COM error code
- **NV_NOT_INITIALIZED** - Initialize was not called
- **NV_UNKNOWN_PROFILE** - no profile loaded
- **NV_INVALID_OPTION** - wrong option constant
- **NV_PROFILE_ERROR** - cannot find option in profile
- **NV_WRONG_OPTION_TYPE** - option is not of type long

### Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files:
- `novaOptions.h`
- `novaOptions.pas`
- `Globals.bas`

For layout options, see Working with Layout objects.

### 1.4.11.4.106 GetLayoutOptionFloat2

The `GetLayoutOptionFloat2` method retrieves an option of float type for a layout object.
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
For layout options, see Working with Layout objects.

1.4.11.4.107 GetLayoutOptionLong

The GetLayoutOptionLong method retrieves an option of long type for a layout object

```c
HRESULT GetLayoutOptionLongString(
    [in, string] LPCWSTR p_pwsObjectId,
    [in, string] LPCWSTR p_pwsLayoutId,
    [in]  LONG  p_nOption,
    [out] LONG* p_plValue);
```

Parameters:
- `p_pwsObjectId` [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_pwsLayoutId` [in] layout id (obtained with GetLayout )
- `p_nOption` [in] option constant
- `p_plValue` [out] will contain the value of the retrieved option

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
For layout options, see Working with Layout objects.

1.4.11.4.108 GetLayoutOptionLong2

The GetLayoutOptionLong2 method retrieves an option of long type for a layout object

```c
HRESULT GetLayoutOptionLong2(
    [in, string] BSTR p_pwsObjectId,
    [in, string] BSTR p_pwsLayoutId,
    [in]  LONG  p_nOption,
    [out] LONG* p_plValue);
```
Parameters:

- `p_pwsObjectId`
  - [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_pwsLayoutId`
  - [in] layout id (obtained with GetLayout)
- `p_nOption`
  - [in] option constant
- `p_plValue`
  - [out] will contain the value of the retrieved option

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

For layout options, see Working with Layout objects.

### 1.4.11.4.109 GetLayoutOptionString

The `GetLayoutOptionString` method retrieves an option of string type for a layout object.

```c
HRESULT GetLayoutOptionString(
    [in, string] LPCWSTR p_pwsObjectId,
    [in, string] LPCWSTR p_pwsLayoutId,
    [in]  LONG    p_nOption,
    [out] LPWSTR* p_pwsValue);
```

Parameters:

- `p_pwsObjectId`
  - [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_pwsLayoutId`
  - [in] layout id (obtained with GetLayout)
- `p_nOption`
  - [in] option constant
- `p_pwsValue`
  - [out] pointer to a pointer to a null terminated Unicode string that will contain the value of the retrieved option.

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
1.4.11.10 GetLayoutOptionString2

The `GetLayoutOptionString2` method retrieves an option of string type for a layout object.

```c
HRESULT GetLayoutOptionString2(
    [in, string] BSTR p_pwsObjectId,
    [in, string] BSTR p_pwsLayoutId,
    [in]  LONG    p_nOption,
    [out] BSTR* p_pwsValue
);```

- **p_pwsObjectId**
  - [in] object id (watermark text, watermark image, overlay, signature or content)
- **p_pwsLayoutId**
  - [in] layout id (obtained with `GetLayout2`)
- **p_nOption**
  - [in] option constant
- **p_pwsValue**
  - [out] pointer to a pointer to a null terminated Unicode string that will contain the value of the retrieved option. On success this value must be freed by the caller with `CoTaskMemFree`.

**Return values:**

- `S_OK` on success or COM error code
- `NV_NOT_INITIALIZED` - Initialize was not called
- `NV_UNKNOWN_PROFILE` - no profile loaded
- `NV_INVALID_OPTION` - wrong option constant
- `NV_PROFILE_ERROR` - cannot find option in profile
- `NV_WRONG_OPTION_TYPE` - option is not of type long

**Remarks:**

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: `novaOptions.h`, `novaOptions.pas`, `Globals.bas`. For layout options, see Working with Layout objects.

1.4.11.11 GetNextProfile

The `GetNextProfile` continues an enumeration of profiles started with `GetFirstProfile`, retrieving the id of the next profile in the enumeration.

```c
HRESULT GetNextProfile(
    [out] LPWSTR* p_pwsProfileId
);```

- **p_pwsProfileId**
  - [out] pointer to a pointer to a null terminated Unicode string containing the name of the next existing profile. The caller must release the allocated memory using `CoTaskMemFree`.

**Return values:**

- `S_OK` on success or COM error code

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: `novaOptions.h`, `novaOptions.pas`, `Globals.bas`. For layout options, see Working with Layout objects.
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_ENUM_NOT_INIT - enumerate was not started, GetFirstProfile was not called before
NV_NO_MORE_PROFILES - there are no more profiles to enumerate

Remarks:
GetNextProfile is used with GetFirstProfile to retrieve profile names.
Sample usage:
hr = GetFirstProfile(&pName);
while (SUCCEEDED(hr) && hr != NV_NO_MORE_PROFILES) {
    // do something with pName
    // ...
    CoTaskMemFree(pName);
    // get next profile if it exists
    hr = GetNextProfile(&pName);
}

1.4.11.4.112 GetNextProfile2

The GetNextProfile2 method continues an enumeration of profiles started with GetFirstProfile, retrieving the id of the next profile in the enumeration.
HRESULT GetNextProfile2(
    [out] BSTR* p_pwsProfileId
);

Parameters:
p_pwsProfileId
    [out] pointer to a pointer to a null terminated Unicode string containing the name of the next profile.

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_ENUM_NOT_INIT - enumerate was not started, GetFirstProfile was not called before
NV_NO_MORE_PROFILES - there are no more profiles to enumerate

1.4.11.4.113 GetOptionBool

The GetOptionBool method retrieves a printing option of boolean type
HRESULT GetOptionBool(
    [in] LONG p_nOption,
    [out] BOOL* p_pbValue
);

Parameters:
p_nOption
    [in] option constant
p_pbValue
    [out] pointer to a boolean that will contain the value of the retrieved option.

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.114 GetOptionEncryptedString

The GetOptionEncryptedString method retrieves an option of type encrypted string

```c
HRESULT GetOptionEncryptedString(
    [in]  LONG  p_nOption,
    [out] LPWSTR* p_pwsValue
);
```

Parameters:
- `p_nOption` [in] option constant
- `p_pwsValue` [out] pointer to a pointer to a null terminated Unicode string that will contain the value of the retrieved option. On success this value must be freed by the caller with CoTaskMemFree.

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.115 GetOptionEncryptedString2

The GetOptionEncryptedString method retrieves an option of type encrypted string

```c
HRESULT GetOptionEncryptedString2(
    [in]  LONG  p_nOption,
    [out] BSTR* p_pwsValue
);
```

Parameters:
- `p_nOption` [in] option constant
- `p_pwsValue` [out] pointer to a pointer to a null terminated Unicode string that will contain the value of the retrieved option. On success this value must be freed by the caller with CoTaskMemFree.
[out] pointer to a pointer to a BSTR string that will contain the value of the retrieved option. On success this value must be freed by the caller with CoTaskMemFree.

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.116 GetOptionLong

The `GetOptionLong` method retrieves a printing option of long int type

```c
HRESULT GetOptionLong(
    [in]  LONG  p_nOption,
    [out] LONG* p_plValue);
```

Parameters:

- `p_nOption` [in] option constant
- `p_plValue` [out] pointer to a long integer that will contain the value of the retrieved option.

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.117 GetOptionString

The `GetOptionString` method retrieves a printing option of string type for a given profile

```c
HRESULT GetOptionString(
    [in]  LONG    p_nOption,
    [out] LPWSTR* p_pwsValue);
```

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
Parameters:

- p_nOption
  [in] option constant

- p_pwsValue
  [out] pointer to a pointer to a null terminated Unicode string that will contain the value of the retrieved option. On success this value must be freed by the caller with CoTaskMemFree.

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.118 GetOptionString2

The **GetOptionString2** method retrieves a printing option of string type for a given profile

```
HRESULT GetOptionString2(
    [in]  LONG  p_nOption,
    [out] BSTR* p_pwsValue
);
```

Parameters:

- p_nOption
  [in] option constant

- p_pwsValue
  [out] return the value of the retrieved option

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.119 GetOverlay

The **GetOverlay** method retrieves the overlay

```
HRESULT GetOverlay(
    [in] LONG  p_nIndex,
    [in] LONG  p_nOption,
    [out] BSTR* p_pwsValue
);
```

Parameters:

- p_nIndex
  [in] index of the overlay

- p_nOption
  [in] option constant

- p_pwsValue
  [out] return the value of the retrieved option

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
[out, string] LPWSTR* p_pwsOverlayId
);

Parameters:
  p_nIndex,
      [in] - overlay index
  p_pwsOverlayId
      [out, string] - overlay id

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_UNKNOWN_PROFILE - no profile loaded
  NV_PROFILE_ERROR - error reading profile

Remarks:
There can be multiple overlays in a profile.
For layout options, see Working with Layout objects.

1.4.11.4.120 GetOverlay2

The GetOverlay2 method retrieves the overlay
  HRESULT GetOverlay2(
      [in] LONG p_nIndex,
      [out, string] BSTR* p_pwsOverlayId
  );

Parameters:
  p_nIndex,
      [in] - overlay index
  p_pwsOverlayId
      [out, string] - overlay id

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_UNKNOWN_PROFILE - no profile loaded
  NV_PROFILE_ERROR - error reading profile

Remarks:
There can be multiple overlays in a profile.
For layout options, see Working with Layout objects.

1.4.11.4.121 GetOverlayCount

The GetOverlayCount method retrieves the number of overlays.
  HRESULT GetOverlayCount(
      [out] LONG* p_pnCount
  );

Parameters:
p pnCount
    [out] count of overlays in profile

Return values:
    S_OK on success or COM error code
    NV_NOT_INITIALIZED - Initialize was not called
    NV_UNKNOWN_PROFILE - no profile loaded

1.4.11.4.122 GetOverlayOptionBool

The GetOverlayOptionBool method retrieves an overlay option of boolean type

HRESULT GetOverlayOptionBool(
    [in, string] LPCWSTR p_wsOverlayId,
    [in]  LONG  p_nOption,
    [out] BOOL* p_pbValue
);

Parameters:
    p_wsOverlayId
        [in] overlay id (obtained with AddOverlay )
    p_nOption
        [in] option constant
    p_pbValue
        [out] pointer to a boolean that will contain the value of the retrieved option

Return values:
    S_OK on success or COM error code
    NV_NOT_INITIALIZED - Initialize was not called
    NV_UNKNOWN_PROFILE - no profile loaded
    NV_INVALID_OVERLAY - wrong overlay id
    NV_INVALID_OPTION - wrong option constant
    NV_PROFILE_ERROR - cannot find option in profile
    NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.123 GetOverlayOptionBool2

The GetOverlayOptionBool2 method retrieves an overlay option of boolean type

HRESULT GetOverlayOptionBool2(
    [in, string] BSTR p_wsOverlayId,
    [in]  LONG  p_nOption,
    [out] BOOL* p_pbValue
);

Parameters:
    p_wsOverlayId
        [in] overlay id (obtained with AddOverlay2 )
p_nOption
    [in] option constant
p_pbValue
    [out] pointer to a boolean that will contain the value of the retrieved option

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OVERLAY - wrong overlay id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.124 GetOverlayOptionEncryptedString

The GetOverlayOptionEncryptedString method retrieves an overlay option of encrypted string type

HRESULT GetOverlayOptionEncryptedString(
    [in, string] LPCWSTR p_wsOverlayId,
    [in]  LONG  p_nOption,
    [out, string] LPWSTR* p_pwsValue
);

Parameters:
p_wsOverlayId
    [in] overlay id (obtained with AddOverlay )
p_nOption
    [in] option constant
p_pwsValue
    [out] pointer to a string that will contain the value of the retrieved option

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OVERLAY - wrong overlay id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
1.4.11.4.125 GetOverlayOptionEncryptedString2

The GetOverlayOptionEncryptedString2 method retrieves an overlay option of encrypted string type

```c
HRESULT GetOverlayOptionEncryptedString2(
    [in, string] BSTR p_wsOverlayId,
    [in]  LONG  p_nOption,
    [out, string] BSTR* p_pwsValue
);
```

Parameters:
- `p_wsOverlayId` [in] overlay id (obtained with AddOverlay )
- `p_nOption` [in] option constant
- `p_pwsValue` [out] pointer to a string that will contain the value of the retrieved option

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OVERLAY - wrong overlay id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.126 GetOverlayOptionFloat

The GetOverlayOptionFloat method retrieves an overlay option of float type

```c
HRESULT GetOverlayOptionFloat(
    [in, string] LPCWSTR p_wsOverlayId,
    [in]  LONG  p_nOption,
    [out] FLOAT* p_pfValue
);
```

Parameters:
- `p_wsOverlayId` [in] overlay id (obtained with AddOverlay )
- `p_nOption` [in] option constant
- `p_pfValue` [out] pointer to a float that will contain the value of the retrieved option

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OVERLAY – wrong overlay id
NV_INVALID_OPTION – wrong option constant
NV_PROFILE_ERROR – cannot find option in profile
NV_WRONG_OPTION_TYPE – option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.127 GetOverlayOptionFloat2

The GetOverlayOptionFloat2 method retrieves an overlay option of float type

HRESULT GetOverlayOptionFloat2(
    [in, string] BSTR p_wsOverlayId,
    [in] LONG  p_nOption,
    [out] FLOAT* p_pfValue
);

Parameters:
- p_wsOverlayId
  - [in] overlay id (obtained with AddOverlay2)
- p_nOption
  - [in] option constant
- p_pfValue
  - [out] pointer to a float that will contain the value of the retrieved option

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED – Initialize was not called
- NV_UNKNOWN_PROFILE – no profile loaded
- NV_INVALID_OVERLAY – wrong overlay id
- NV_INVALID_OPTION – wrong option constant
- NV_PROFILE_ERROR – cannot find option in profile
- NV_WRONG_OPTION_TYPE – option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.128 GetOverlayOptionLong

The GetOverlayOptionLong method retrieves an overlay option of boolean type

HRESULT GetOverlayOptionLong(
    [in, string] LPCWSTR p_wsOverlayId,
    [in] LONG  p_nOption,
    [out] LONG* p_plValue
);

Parameters:
- p_wsOverlayId
  - [in] overlay id (obtained with AddOverlay)
The `GetOverlayOptionLong2` method retrieves an overlay option of long type

```c
HRESULT GetOverlayOptionLong2(
    [in, string] BSTR p_wsOverlayId,
    [in]  LONG  p_nOption,
    [out] LONG* p_plValue
);
```

**Parameters:**
- `p_wsOverlayId` [in] overlay id (obtained with AddOverlay2)
- `p_nOption` [in] option constant
- `p_plValue` [out] pointer to a long that will contain the value of the retrieved option

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OVERLAY - wrong overlay id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
1.4.11.4.130 GetOverlayOptionString

The GetOverlayOptionString method retrieves an overlay option of string type

```c
HRESULT GetOverlayOptionString(
    [in, string] LPCWSTR p_wsOverlayId,
    [in]   LONG  p_nOption,
    [out, string] LPWSTR* p_pwsValue
);
```

**Parameters:**
- `p_wsOverlayId` [in] overlay id (obtained with AddOverlay)
- `p_nOption` [in] option constant
- `p_pwsValue` [out] pointer to a string that will contain the value of the retrieved option

**Return values:**
- `S_OK` on success or COM error code
- `NV_NOT_INITIALIZED` - Initialize was not called
- `NV_UNKNOWN_PROFILE` - no profile loaded
- `NV_INVALID_OVERLAY` - wrong overlay id
- `NV_INVALID_OPTION` - wrong option constant
- `NV_PROFILE_ERROR` - cannot find option in profile
- `NV_WRONG_OPTION_TYPE` - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.131 GetOverlayOptionString2

The GetOverlayOptionString2 method retrieves an overlay option of string type

```c
HRESULT GetOverlayOptionString2(
    [in, string] BSTR p_wsOverlayId,
    [in]   LONG  p_nOption,
    [out, string] BSTR* p_pwsValue
);
```

**Parameters:**
- `p_wsOverlayId` [in] overlay id (obtained with AddOverlay2)
- `p_nOption` [in] option constant
- `p_pwsValue` [out] pointer to a string that will contain the value of the retrieved option

**Return values:**
- `S_OK` on success or COM error code
- `NV_NOT_INITIALIZED` - Initialize was not called
- `NV_UNKNOWN_PROFILE` - no profile loaded
NovaPDF SDK

1.4.11.4.132  GetOverrideOptionBool

The GetOverrideOptionBool method gets an override printing option of boolean type

```c
HRESULT GetOverrideOptionBool(
    [in] LONG p_nOption,
    [out] BOOL* p_pbValue);
```

**Parameters:**

- `p_nOption` [in] option constant
- `p_pbValue` [out] bool value

**Return values:**

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - wrong option constant

**Remarks:**

You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.133  GetOverrideOptionEncryptedString

The GetOverrideOptionEncryptedString method gets an override printing option of encrypted string type

```c
HRESULT GetOverrideOptionEncryptedString(
    [in] LONG p_nOption,
    [out, string] LPWSTR* p_pwsValue);
```

**Parameters:**

- `p_nOption` [in] option constant
- `p_pwsValue` [out] string value

**Return values:**

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - wrong option constant

**Remarks:**

You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas. Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.134 GetOverrideOptionEncryptedString2

The GetOverrideOptionEncryptedString2 method gets an override printing option of encrypted string type

```c
HRESULT GetOverrideOptionEncryptedString2(
    [in] LONG p_nOption,
    [out, string] BSTR* p_pwsValue
);
```

Parameters:
- p_nOption [in] option constant
- p_pwsValue [out] string value

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - wrong option constant

Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas. Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.135 GetOverrideOptionLong

The GetOverrideOptionLong method gets an override printing option of long type

```c
HRESULT GetOverrideOptionLong(
    [in] LONG p_nOption,
    [out] LONG* p_pValue
);
```

Parameters:
- p_nOption [in] option constant
- p_pValue [out] long integer value

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - wrong option constant
Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas. Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.136 GetOverrideOptionString

The GetOverrideOptionString method gets an override printing option of string type

```cpp
HRESULT GetOverrideOptionString(
    [in] LONG    p_nOption,
    [out, string] LPWSTR*  p_pwsValue
);
```

Parameters:
- `p_nOption` [in] option constant
- `p_pwsValue` [out] string value

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - wrong option constant

Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas. Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.137 GetOverrideOptionString2

The GetOverrideOptionString2 method gets an override printing option of string type

```cpp
HRESULT GetOverrideOptionString2(
    [in] LONG    p_nOption,
    [out, string] BSTR*  p_pwsValue
);
```

Parameters:
- `p_nOption` [in] option constant
- `p_pwsValue` [out] string value

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - wrong option constant

Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder
of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.138  GetPDFFileName

The GetPDFFileName method retrieves the name of the last generated PDF file.

```cpp
HRESULT GetPDFFileName(
    [in] BOOL p_bPrintStarted,
    [out, string] LPWSTR* p_pwsFileName);
```

Parameters:
- `p_bPrintStarted` [in] flag, what file name to retrieve. See Remarks.
- `p_pwsFileName` [out] pointer to a pointer to a null terminated Unicode string that will contain the file name. On success this value must be freed by the caller with CoTaskMemFree.

Return values:
- S_OK on success or COM error code
- NO_INIT - Initialize was not called

Remarks:
Because the novaPDF SDK 11 works with the printer spooler queue, the documents sent to the printer are added to the queue. If there are already some other documents in the queue, the current document is not processed until the previous ones are finished.

There are two PDF file names you can find out, depending on the value of the `p_bPrintStarted` flag:
- the name of the PDF file that was just sent to the printer
- the name of the PDF file that is currently processed by the printer

This information is available only on the computer that starts the print job, if the PDF file is saved local and not on the network.

1.4.11.4.139  GetPDFFileName2

The GetPDFFileName2 method retrieves the name of the last generated PDF file.

```cpp
HRESULT GetPDFFileName2(
    [in] BOOL p_bPrintStarted,
    [out, string] BSTR* p_pwsFileName);
```

Parameters:
- `p_bPrintStarted` [in] flag, what file name to retrieve. See Remarks.
- `p_pwsFileName` [out] will contain the file name.
Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called

Remarks:
Because the novaPDF SDK 11 works with the printer spooler queue, the documents sent to the printer are added to the queue. If there are already some other documents in the queue, the current document is not processed until the previous ones are finished.

There are two PDF file names you can find out, depending on the value of the p_bPrintStarted flag:
  • the name of the PDF file that was just sent to the printer
  • the name of the PDF file that is currently processed by the printer

This information is available only on the computer that starts the print job, if the PDF file is saved local and not on the network.

1.4.11.4.140 GetSaveRule

The GetSaveRule method returns the save rule for the given index

HRESULT GetSaveRule(
    [in] LONG p_nIndex,
    [out] LONG* p_nType,
    [out, string] LPWSTR* p_pwsName,
    [out, string] LPWSTR* p_pwsDescription,
    [out, string] LPWSTR* p_pwsText,
    [out, string] LPWSTR* p_pwsFormat,
    [out] BOOL* p_pbEnabled
);

Parameters:
  p_nIndex
    [in] save rule index
  p_nType
    [out] save rule type (0-remove from start, 1- remove from end, 2 - remove all,
  p_pwsName
    [out] rule name
  p_pwsDescription
    [out] rule description
  p_pwsText
    [out] text to remove or regular expression
  p_pwsFormat
    [out] format for regular expression
  p_pbEnabled
    [out] flag, tag is enabled

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_UNKNOWN_PROFILE - no profile loaded
1.4.11.4.141 GetSaveRule2

The `GetSaveRule2` method returns the save rule for the given index.

```c
HRESULT GetSaveRule2(
    [in] LONG p_nIndex,
    [out] LONG* p_pnType,
    [out, string] BSTR* p_pwsName,
    [out, string] BSTR* p_pwsDescription,
    [out, string] BSTR* p_pwsText,
    [out, string] BSTR* p_pwsFormat,
    [out] BOOL* p_pbEnabled
);
```

**Parameters:**

- `p_nIndex`: [in] save rule index
- `p_pnType`: [out] save rule type (0- remove from start, 1- remove from end, 2 - remove all),
- `p_pwsName`: [out] rule name
- `p_pwsDescription`: [out] rule description
- `p_pwsText`: [out] text to remove or regular expression
- `p_pwsFormat`: [out] format for regular expression
- `p_pbEnabled`: [out] flag, tag is enabled

**Return values:**

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_SAVE_RULE - invalid save rule index

**Remarks:**

This method retrieves information about the save rule at given index.

1.4.11.4.142 GetSaveRulesCount

The `GetSaveRulesCount` method returns save rules count.

```c
HRESULT GetSaveRulesCount(
    [out] LONG* p_pnCount
);
```
Parameters:

p_pnCount
  [in] tags count

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded

1.4.11.4.143 GetSignature

The GetSignature method retrieves the signature and its layout object

HRESULT GetSignature(
    [out, string] LPWSTR* p_pwsSignatureId
    [out, string] LPWSTR* p_pwsLayoutId
);

Parameters:

p_pwsSignatureId
  [out, string] - signature id
p_pwsLayoutId
  [out, string] - layout id

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_PROFILE_ERROR - error reading profile

Remarks:
There can be only one signature in a profile.
For layout options, see Working with Layout objects.

1.4.11.4.144 GetSignature2

The GetSignature2 method retrieves the signature and its layout object

HRESULT GetSignature2(
    [out, string] BSTR* p_pwsSignatureId
    [out, string] BSTR* p_pwsLayoutId
);

Parameters:

p_pwsSignatureId
  [out, string] - signature id
p_pwsLayoutId
  [out, string] - layout id

Return values:

S_OK on success or COM error code
Remarks:
There can be only one signature in a profile.
For layout options, see Working with Layout objects.

1.4.11.4.145 GetUserTag

The **GetUserTag** method returns the user tag for the given index

```c
HRESULT GetUserTag(
    [in] LONG p_nIndex,
    [out, string] LPWSTR* p_pwsTag,
    [out, string] LPWSTR* p_pwsDefaultValue,
    [out] BOOL* p_pbEnabled
);
```

**Parameters:**
- **p_nIndex**
  - [in] tag index
- **p_wsTag**
  - [out] user tag
- **p_wsDefaultValue**
  - [out] default tag value
- **p_bEnabled**
  - [out] flag, tag is enabled

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_USER_TAG - invalid user tag index

**Remarks:**
This method returns information about the tag at position given by p_nIndex

1.4.11.4.146 GetUserTag2

The **GetUserTag2** method returns the user tag for the given index

```c
HRESULT GetUserTag2(
    [in] LONG p_nIndex,
    [out, string] BSTR* p_pwsTag,
    [out, string] BSTR* p_pwsDefaultValue,
    [out] BOOL* p_pbEnabled
);
```

**Parameters:**
- **p_nIndex**
  - [in] tag index
p_wsTag  
[out] user tag
p_wsDefaultValue
[out] default tag value
p_bEnabled
[out] flag, tag is enabled

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_USER_TAG - invalid user tag index

Remarks:
This method returns information about the tag at position given by p_nIndex

1.4.11.4.147 GetUserTagsCount

The GetUserTagsCount method returns user tags count

HRESULT GetUserTagsCount(
    [out] LONG* p_pnCount
);

Parameters:
p_pnCount
    [in] tags count

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded

1.4.11.4.148 GetWatermarkImage

The GetWatermarkImage method retrieves an existing image watermark properties.

HRESULT GetWatermarkImage(
    [in] LONG p_nIndex,
    [out, string] LPWSTR* p_pwsWatermarkId
);

Parameters:
p_nIndex,
    [in] - watermark index
p_pwsWatermarkId
    [out, string] - watermark id

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
1.4.11.4.149 GetWatermarkImage2

The GetWatermarkImage2 method retrieves an existing image watermark properties.

```c
HRESULT GetWatermarkImage2([in] LONG p_nIndex, [out, string] BSTR* p_pwsWatermarkId);
```

**Parameters:**
- `p_nIndex`: [in] - watermark index
- `p_pwsWatermarkId`: [out, string] - watermark id

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong image watermark index

1.4.11.4.150 GetWatermarkImageCount

The GetWatermarkImageCount method retrieves the number of image watermarks.

```c
HRESULT GetWatermarkImageCount([out] SHORT* p_pnCount);
```

**Parameters:**
- `p_pnCount`: [out] count of image watermarks

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded

1.4.11.4.151 GetWatermarkImageOptionBool

The GetWatermarkImageOptionBool method retrieves a watermark image option of boolean type.

```c
HRESULT GetWatermarkImageOptionBool([in, string] LPCWSTR p_pwsWatermarkId, [in]  LONG  p_nOption, [out] BOOL* p_pbValue);
```

**Parameters:**
- `p_pwsWatermarkId`:
[in] watermark id (obtained with GetWatermarkImage )
p_nOption
  [in] option constant
p_pbValue
  [out] pointer to a boolean that will contain the value of the retrieved option

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_IMG - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.152 GetWatermarkImageOptionBool2

The GetWatermarkImageOptionBool2 method retrieves a watermark image option of boolean type

HRESULT GetWatermarkImageOptionBool2(
    [in, string] BSTR p_pwsWatermarkId,
    [in]  LONG  p_nOption,
    [out] BOOL* p_pbValue
);

Parameters:
p_pwsWatermarkId
  [in] watermark id (obtained with GetWatermarkImage2 )
p_nOption
  [in] option constant
p_pbValue
  [out] pointer to a boolean that will contain the value of the retrieved option

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_IMG - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
1.4.11.4.153 GetWatermarkImageOptionEncryptedString

The GetWatermarkImageOptionEncryptedString method retrieves a watermark image option of encrypted string type

```c
HRESULT GetWatermarkImageOptionEncryptedString(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in]    LONG    p_nOption,
    [out, string] LPWSTR* p_pwsValue
);
```

**Parameters:**
- `p_pwsWatermarkId` [in] watermark id (obtained with GetWatermarkImage)
- `p_nOption` [in] option constant
- `p_pwsValue` [out] pointer to a pointer to a null terminated Unicode string that will contain the value of the retrieved option. On success this value must be freed by the caller with CoTaskMemFree.

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.154 GetWatermarkImageOptionEncryptedString2

The GetWatermarkImageOptionEncryptedString2 method retrieves a watermark image option of encrypted string type

```c
HRESULT GetWatermarkImageOptionEncryptedString2(
    [in, string] BSTR p_pwsWatermarkId,
    [in]    LONG    p_nOption,
    [out, string] BSTR* p_pwsValue
);
```

**Parameters:**
- `p_pwsWatermarkId` [in] watermark id (obtained with GetWatermarkImage2)
- `p_nOption` [in] option constant
- `p_pwsValue` [out] pointer to a pointer to a null terminated Unicode string that will contain the value of the retrieved option.

**Return values:**
- S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_IMG - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.155 GetWatermarkImageOptionFloat

The GetWatermarkImageOptionFloat method retrieves a watermark image option of float type

```
HRESULT GetWatermarkImageOptionFloat(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in]  LONG   p_nOption,
    [out] FLOAT* p_pfValue);
```

Parameters:

- `p_pwsWatermarkId` [in] watermark id (obtained with GetWatermarkImage)
- `p_nOption` [in] option constant
- `p_pfValue` [out] pointer to a pointer to a null terminated Unicode string that will contain the value of the retrieved option. On success this value must be freed by the caller with CoTaskMemFree.

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.156 GetWatermarkImageOptionFloat2

The GetWatermarkImageOptionFloat2 method retrieves a watermark image option of boolean type

```
HRESULT GetWatermarkImageOptionFloat2(
    [in, string] BSTR p_pwsWatermarkId,
    [in]  LONG  p_nOption,
    [out] FLOAT* p_pfValue);
```

Parameters:
p_pwsWatermarkId
    [in] watermark id (obtained with GetWatermarkImage2 )
p_nOption
    [in] option constant
p_plValue
    [out] the value of the retrieved option

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_IMG - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files:
novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.157 GetWatermarkImageOptionLong
The GetWatermarkImageOptionLong method retrieves a watermark image option of long int type

HRESULT GetWatermarkImageOptionLong(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in]  LONG  p_nOption,
    [out] LONG* p_plValue);

Parameters:
    p_pwsWatermarkId
    [in] watermark id (obtained with GetWatermarkImage )
    p_nOption
    [in] option constant
    p_plValue
    [out] pointer to a pointer to a null terminated Unicode string that will contain

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_IMG - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files:
novaOptions.h, novaOptions.pas, Globals.bas.
1.4.11.4.158  GetWatermarkImageOptionLong2

The **GetWatermarkImageOptionLong2** method retrieves a watermark image option of long type

```c
HRESULT GetWatermarkImageOptionLong2(
    [in, string] BSTR p_pwsWatermarkId,
    [in]    LONG  p_nOption,
    [out]    LONG* p_plValue
);
```

**Parameters:**
- `p_pwsWatermarkId`: [in] watermark id (obtained with GetWatermarkImage2)
- `p_nOption`: [in] option constant
- `p_plValue`: [out] the value of the retrieved option

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.159  GetWatermarkImageOptionString

The **GetWatermarkImageOptionString** method retrieves a watermark image option of string type

```c
HRESULT GetWatermarkImageOptionString(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in]    LONG    p_nOption,
    [out, string] LPWSTR* p_pwsValue
);
```

**Parameters:**
- `p_pwsWatermarkId`: [in] watermark id (obtained with GetWatermarkImage)
- `p_nOption`: [in] option constant
- `p_pwsValue`: [out] pointer to a pointer to a null terminated Unicode string that will contain

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_IMG - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.160 GetWatermarkImageOptionString2

The GetWatermarkImageOptionString2 method retrieves a watermark image option of string type

```
HRESULT GetWatermarkImageOptionString2(
    [in, string] BSTR p_pwsWatermarkId,
    [in] LONG  p_nOption,
    [out, string] BSTR* p_pwsValue
);
```

Parameters:
- p_pwsWatermarkId
  [in] watermark id (obtained with GetWatermarkImage2)
- p_nOption
  [in] option constant
- p_pwsValue
  [out] the value of the retrieved option

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.161 GetWatermarkTextOptionBool

The GetWatermarkTextOptionBool method retrieves a watermark text option of boolean type

```
HRESULT GetWatermarkTextOptionBool(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in] LONG  p_nOption,
    [out] BOOL* p_pbValue
);
```

Parameters:
- p_pwsWatermarkId
  [in] watermark id (obtained with GetWatermarkText)
p_nOption
  [in] option constant
p_plValue
  [out] pointer to a pointer to a null terminated Unicode string that will contain the value of the retrieved option. On success this value must be freed by the caller with CoTaskMemFree.

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_TXT - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.162 GetWatermarkTextOptionBool2

The GetWatermarkTextOptionBool2 method retrieves a watermark text option of boolean type

HRESULT GetWatermarkTextOptionBool2(
  [in, string] BSTR p_pwsWatermarkId,
  [in]  LONG  p_nOption,
  [out] BOOL* p_pbValue);

Parameters:
p_pwsWatermarkId
  [in] watermark id (obtained with GetWatermarkText2 )
p_nOption
  [in] option constant
p_pbValue
  [out] the value of the retrieved option

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_TXT - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
### GetWatermarkTextOptionFloat

The `GetWatermarkTextOptionFloat` method retrieves a watermark text option of float type.

```c
HRESULT GetWatermarkTextOptionFloat(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in]   LONG   p_nOption,
    [out] FLOAT* p_pfValue);
```

**Parameters:**
- `p_pwsWatermarkId` [in] watermark id (obtained with GetWatermarkText)
- `p_nOption` [in] option constant
- `p_pfValue` [out] pointer to a pointer to a null terminated Unicode string that will contain the value of the retrieved option. On success this value must be freed by the caller with CoTaskMemFree.

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_TXT - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

### GetWatermarkTextOptionFloat2

The `GetWatermarkTextOptionFloat2` method retrieves a watermark text option of boolean type.

```c
HRESULT GetWatermarkTextOptionFloat2(
    [in, string] BSTR p_pwsWatermarkId,
    [in]   LONG  p_nOption,
    [out] FLOAT* p_pfValue);
```

**Parameters:**
- `p_pwsWatermarkId` [in] watermark id (obtained with GetWatermarkText2)
- `p_nOption` [in] option constant
- `p_pfValue` [out] the value of the retrieved option

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_TXT - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files:
novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.165 GetWatermarkTextOptionLong

The GetWatermarkTextOptionLong method retrieves a watermark text option of long int type

HRESULT GetWatermarkTextOptionLong(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in]  LONG  p_nOption,
    [out] LONG* p_plValue);

Parameters:
p_pwsWatermarkId
    [in] watermark id (obtained with GetWatermarkText )
p_nOption
    [in] option constant
p_plValue
    [out] pointer to a pointer to a null terminated Unicode string that will contain

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_TXT - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files:
novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.166 GetWatermarkTextOptionLong2

The GetWatermarkTextOptionLong2 method retrieves a watermark text option of long type

HRESULT GetWatermarkTextOptionLong2(
    [in, string] BSTR p_pwsWatermarkId,
    [in]  LONG  p_nOption,
    [out] LONG* p_plValue);

Parameters:
p_pwsWatermarkId
    [in] watermark id (obtained with GetWatermarkText2 )
p_nOption
  [in] option constant
p_pValue
  [out] the value of the retrieved option

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_TXT - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files:
novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.167 GetWatermarkTextOptionString

The GetWatermarkTextOptionString method retrieves a watermark text option of string type

HRESULT GetWatermarkTextOptionString(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in]  LONG    p_nOption,
    [out, string] LPWSTR* p_pwsValue
);

Parameters:

p_pwsWatermarkId
  [in] watermark id (obtained with GetWatermarkText )
p_nOption
  [in] option constant
p_pwsValue
  [out] pointer to a pointer to a null terminated Unicode string that will contain...
1.4.11.4.168 GetWatermarkTextOptionString2

The GetWatermarkTextOptionString2 method retrieves a watermark text option of string type

```c
HRESULT GetWatermarkTextOptionString2(
    [in, string] BSTR p_pwsWatermarkId,
    [in]  LONG  p_nOption,
    [out, string] BSTR* p_pwsValue
);
```

**Parameters:**
- `p_pwsWatermarkId` [in] watermark id (obtained with GetWatermarkText2 )
- `p_nOption` [in] option constant
- `p_pwsValue` [out] the value of the retrieved option

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_TXT - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.169 GetWatermarkText

The GetWatermarkText method retrieves an existing image watermark properties.

```c
HRESULT GetWatermarkText(
    [in] LONG p_nIndex,
    [out, string] LPWSTR* p_pwsWatermarkId
);
```

**Parameters:**
- `p_nIndex`, [in] - watermark index
- `p_pwsWatermarkId` [out, string] - watermark id

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong image watermark index
1.4.11.4.170 GetWatermarkText2

The `GetWatermarkText2` method retrieves an existing image watermark properties.

```c
HRESULT GetWatermarkText2(
    [in] LONG p_nIndex,
    [out, string] BSTR* p_pwsWatermarkId
);
```

Parameters:
- `p_nIndex`, [in] - watermark index
- `p_pwsWatermarkId`, [out, string] - watermark id

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong image watermark index

1.4.11.4.171 GetWatermarkTextCount

The `GetWatermarkTextCount` method retrieves the number of image watermarks.

```c
HRESULT GetWatermarkTextCount(
    [out] SHORT* p_pnCount
);
```

Parameters:
- `p_pnCount`, [out] - count of image watermarks

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded

1.4.11.4.172 Initialize

The `Initialize` method initializes the `INovaPdfOptions` interface.

```c
HRESULT Initialize(
    [in] LPCWSTR p_wsPrinterName,
    [in] LPCWSTR p_wsLicenseKey
);
```

Parameters:
- `p_wsPrinterName`, [in] - pointer to a null terminated Unicode string containing the name of the printer
- `p_wsLicenseKey`, [in] - pointer to a null terminated Unicode string containing the license key

Return values:
- S_OK on success or COM error code
- NV_INVALID_PRINTER_NAME - cannot find printer with given printer name
NV_NOT_A_NOVAPDF_PRINTER - printer is not a novaPDF SDK 11
NV_INVALID_LICENSE - cannot read license or not a SDK license
NV_SERVICE_ERROR - cannot connect to novaPDF Server service

Remarks:
This method must be called prior to calling any method from the INovaPdfOptions interface.

1.4.11.4.173  Initialize2

The Initialize2 method initializes the INovaPdfOptions interface

```c
HRESULT Initialize2(
    [in] BSTR p_wsPrinterName,
    [in] BSTR p_wsLicenseKey
);
```

Parameters:
- `p_wsPrinterName` [in] printer name
- `p_wsLicenseKey` [in] license key

Return values:
- S_OK on success or COM error code
- NV_INVALID_PRINTER_NAME - cannot find printer with given printer name
- NV_NOT_A_NOVAPDF_PRINTER - printer is not a novaPDF SDK 11
- NV_INVALID_LICENSE - cannot read license or not a SDK license
- NV_SERVICE_ERROR - cannot connect to novaPDF Server service

Remarks:
This method must be called prior to calling any method from the INovaPdfOptions interface.

1.4.11.4.174  InitializeOLEUsage

The InitializeOLEUsage method initializes the OLE server licensing

```c
HRESULT InitializeOLEUsage(
    [in] BSTR p_pwstrOLEProgID,);
```

Parameters:
- `p_pwstrOLEProgID` [in] pointer to a Unicode string containing the ProgID for the OLE server that

Return values:
- S_OK on success or COM error code

Remarks:
This method must be called prior to initializing the OLE object that will perform the print to the
novaPDF SDK 11.

1.4.11.4.175  InitializeSilent

The InitializeSilent method initializes the INovaPdfOptions interface silently

```c
HRESULT InitializeSilent(
    [in] LPCWSTR p_wsPrinterName,
    [in] LPCWSTR p_wsLicenseKey
);
Parameters:

- `p_wsPrinterName` [in] pointer to a null terminated Unicode string containing the name of the printer to configure
- `p_wsLicenseKey` [in] pointer to a null terminated Unicode string containing the license key

Return values:

- S_OK on success or COM error code
- NV_INVALID_PRINTER_NAME - cannot find printer with given printer name
- NV_NOT_A_NOVAPDF_PRINTER - printer is not a novaPDF SDK 11
- NV_INVALID_LICENSE - cannot read license or not a SDK license
- NV_SERVICE_ERROR - cannot connect to novaPDF Server service

Remarks:

This method can be used instead of the Initialize method, when you don't want to have message boxes shown with error messages. Use it when your application runs as a windows service or on a server computer.

1.4.11.4.176 InitializeSilent2

The `InitializeSilent2` method initializes the INovaPdfOptions interface silently

```c
HRESULT InitializeSilent2(
    [in] BSTR p_wsPrinterName,
    [in] BSTR p_wsLicenseKey
);
```

Parameters:

- `p_wsPrinterName` [in] pointer to a BSTR containing the name of the printer to configure
- `p_wsLicenseKey` [in] pointer to a BSTR containing the license key

Return values:

- S_OK on success or COM error code
- NV_INVALID_PRINTER_NAME - cannot find printer with given printer name
- NV_NOT_A_NOVAPDF_PRINTER - printer is not a novaPDF SDK 11
- NV_INVALID_LICENSE - cannot read license or not a SDK license
- NV_SERVICE_ERROR - cannot connect to novaPDF Server service

Remarks:

This method can be used instead of the Initialize method, when you don't want to have message boxes shown with error messages. Use it when your application runs as a windows service or on a server computer.

1.4.11.4.177 LicenseApplication

The `LicenseApplication` method licences an application to print to novaPDF

```c
HRESULT LicenseApplication(
    [in] BSTR p_pwstrAppName,
);
```
Parameters:

\texttt{p\_pwstrAppName}
\begin{itemize}
  \item \texttt{[in]} pointer to a Unicode string containing the name of the application that will be executed
\end{itemize}

Return values:

\texttt{S\_OK} on success or COM error code

Remarks:

This method must be called prior to launching the application with the specified name. This call assures that the application will print without the notice on bottom of pages.

1.4.11.4.178 LicenseOLEServer

The \texttt{LicenseOLEServer} method license the OLE server prior initialized with \texttt{InitializeOLEServer}

\begin{verbatim}
HRESULT LicenseOLEServer(void);
\end{verbatim}

Return values:

\texttt{S\_OK} on success or COM error code

Remarks:

This method must be called after initializing the OLE object that will perform the print to the novaPDF SDK 11

1.4.11.4.179 LicenseShellExecuteFile

The \texttt{LicenseShellExecuteFile} method licences a document to be printed with \texttt{ShellExecute}

\begin{verbatim}
HRESULT LicenseShellExecuteFile(
    [in] BSTR p\_pwstrFileName,
);
\end{verbatim}

Parameters:

\begin{itemize}
  \item \texttt{p\_pwstrFileName}
    \begin{itemize}
      \item \texttt{[in]} pointer to a Unicode string containing the name of the file that will be licensed
    \end{itemize}
\end{itemize}

Return values:

\texttt{S\_OK} on success or COM error code

Remarks:

This method must be called prior to calling the \texttt{ShellExecute} function for the given parameter. This call assures that the document will be printed without the notice on bottom of pages, even if the application that prints the document is already opened.

1.4.11.4.180 LoadProfile

The \texttt{LoadProfile} method loads an existing profile.

\begin{verbatim}
HRESULT LoadProfile(
    [in] LPCWSTR p\_wsProfileId
);
\end{verbatim}

Parameters:

\begin{itemize}
  \item \texttt{p\_wsProfileId}
    \begin{itemize}
      \item \texttt{[in]} pointer to a null terminated Unicode string containing the id of the profile
    \end{itemize}
\end{itemize}
Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_SERVICE_ERROR - error connecting to novaPDF Server service
NV_PROFILE_ERROR - error reading profiles

1.4.11.4.181 LoadProfile2

The LoadProfile2 method loads an existing profile.

HRESULT LoadProfile2(
    [in] BSTR p_wsProfileId
);

Parameters:
p_wsProfileId
    [in] id of the profile to load

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_SERVICE_ERROR - error connecting to novaPDF Server service
NV_PROFILE_ERROR - error reading profiles

1.4.11.4.182 ModifyBookmarkDefinition

The ModifyBookmarkDefinition method modifies an existing bookmark definition, having the characteristics specified by the method parameters.

HRESULT ModifyBookmarkDefinition(
    [in] SHORT p_nDefinition,
    [in] SHORT p_nHeading,
    [in] BOOL p_bEnabled,
    [in] BOOL p_bDetFont,
    [in, string] LPCWSTR p_wsDetFont,
    [in] BOOL p_bDetStyle,
    [in] BOOL p_bDetBold,
    [in] BOOL p_bDetItalic,
    [in] BOOL p_bDetSize,
    [in] FLOAT p_nDetSizeVal,
    [in] FLOAT p_nDetSizePt,
    [in] BOOL p_bDispAsBold,
    [in] BOOL p_bDispAsItalic,
    [in] LONG p_nDispColor
);

Parameters:
p_nDefinition
    [in] definition index
p_nHeading
    [in] heading index
p_bEnabled
    [in] definition is enabled
p_bDetFont
    [in] detect font flag
1.4.11.4.183 ModifyBookmarkDefinition2

The ModifyBookmarkDefinition2 method modifies an existing bookmark definition, having the characteristics specified by the method parameters.

```
HRESULT ModifyBookmarkDefinition2(
    [in] SHORT p_nDefinition,
    [in] SHORT p_nHeading,
    [in] BOOL p_bEnabled,
    [in] BOOL p_bDetFont,
    [in, string] BSTR p_wsDetFont,
    [in] BOOL p_bDetStyle,
    [in] BOOL p_bDetBold,
    [in] BOOL p_bDetItalic,
    [in] BOOL p_bDispAsBold,
    [in] BOOL p_bDispAsItalic,
    [in] LONG p_nDispColor,
    [in, string] BSTR p_wsProfileName,
    [in] shorts p_nDispSizeVal,
    [in] shorts p_nDispSizePt,
    [in] BOOL p_bDispColor,
    [in] shorts p_nDispColor,
    [in, string] BSTR p_wsDispProfileName
);```

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_BOOKMARK_DEF - wrong bookmark definition index
[in] BOOL p_bPublicProfile
);

Parameters:

p_nDefinition
  [in] definition index
p_nHeading
  [in] heading index
p_bEnabled
  [in] definition is enabled
p_bDetFont
  [in] detect font flag
p_wsDetFont
  [in] font name
p_bDetStyle
  [in] detect font style
p_bDetBold
  [in] bold font
p_bDetItalic
  [in] italic font
p_bDetSize
  [in] detect font size
p_nDetSizeVal
  [in] font size
p_nDetSizePt
  [in] font size rounding
p_bDetColor
  [in] detect font color
p_nDetColor
  [in] font color (RGB value)
p_bDispAsBold
  [in] display bookmark font bold
p_bDispAsItalic
  [in] display bookmark font italic
p_nDispColor
  [in] display bookmark font color
p_wszProfileName
  [in] pointer to a BSTR containing the profile to modify. If this parameter is a
  [in] Flag if the profile is a public or a private profile.

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED – Initialize was not called
  NV_UNKNOWN_PROFILE – no profile loaded
  NV_INVALID_BOOKMARK_DEF – wrong bookmark definition index

1.4.11.4.184 RegisterEventWindow

The RegisterEventWindow registers a window with the printer in order to receive printing messages.

HRESULT RegisterEventWindow(
  [in] LONG p_hWnd
);

Parameters:

p_hWnd
[in] handle to the window (cast to a LONG value), that will receive the printer messages.

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called

1.4.11.4.185 RegisterNovaEvent

The RegisterNovaEvent registers a Windows event that will be signaled by the printer

```c
HRESULT RegisterNovaEvent([in] LPCWSTR p_wsEventName);
```

Parameters:
  p_wsEventName
  [in] Name of the event. See How to use events topic for a list of possible event names.

Return values:
  S_OK - on success
  S_FALSE - event cannot be created

1.4.11.4.186 RegisterNovaEvent2

The RegisterNovaEvent2 registers a Windows event that will be signaled by the printer

```c
HRESULT RegisterNovaEvent2([in] BSTR p_wsEventName);
```

Parameters:
  p_wsEventName
  [in] Name of the event. See How to use events topic for a list of possible event names.

Return values:
  S_OK - on success
  S_FALSE - event cannot be created

1.4.11.4.187 RestoreDefaultPrinter

The RestoreDefaultPrinter method restores the default printer to the printer that was default before calling SetDefaultPrinter.

```c
HRESULT RestoreDefaultPrinter(void);
```

Parameters:
  none

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_NODEFAULT_PRINTER - SetDefaultPrinter was not called

Remarks:
After calling SetDefaultPrinter with an INovaPdfOptions object, call RestoreDefaultPrinter with the same object to restore the original default printer.
1.4.11.4.188  SaveProfile

The `SaveProfile` method loads an existing profile.

```c
HRESULT SaveProfile(void);
```

Parameters:
none

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_SERVICE_ERROR - error connecting to novaPDF Server service
- NV_NO_PROFILE - no profile loaded
- NV_PROFILE_ERROR - error reading profiles
- NV_PROFILE_SAVE_ERROR - error saving profile

1.4.11.4.189  SelectGMailAccount

The `SelectGMailAccount` starts a GMail OAuth authentication process to select the GMail account for sending OAuth email.

```c
HRESULT SelectGMailAccount(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG p_nAccountType
);
```

Parameters:
- p_pwsActionId
  - [in] Action id
- p_nAccountType
  - [in] oauth gmail type constant, can be OAUTH_GMAILSAME_ACCOUNT or OAUTH_GMAIL_REUSE_ACCOUNT

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION  - wrong action id
- NV_INVALID_OPTION  - wrong oauth gmail type constant for p_nAccountType
- NV_ERROR_GMAIL     - GMail authentication failed

Remarks:
This function will select the GMail account for current loaded profile and provided action id. The account will be saved for the profile only when calling SaveProfile.

GMail OAuth authentication starts the default Internet browser where you can enter the GMail account and authenticate.

There are three modes that can be used with GMail OAuth email type:
- OAUTH_GMAILSAME_ACCOUNT: all emails will be sent with the same account, for all users that use this profile. The credentials are saved with the profile and reused for emails. The GMail account can be selected from Profile Manger or calling SelectGMailAccount. Credentials are not asked when the email is sent.
- OAUTH_GMAIL_ASK_ACCOUNT: GMail credentials are not saved with the profile. Each time an email is sent the user will be asked to select the GMail account and
authenticate.
- OAUTH_GMAIL_REUSE_ACCOUNT: GMail credentials are saved with the profile, but they are different for each user that uses the profile. User will be asked to select account and authenticate when the first email is sent then the credentials are reused for the same user. The GMail Account can also be selected or changed from the Save As dialog - Recipients dialog or calling SelectGMailAccount.

Take care that if you wish to reuse GMail account credentials with one of the OAUTH_GMAIL_SAME_ACCOUNT or OAUTH_GMAIL_REUSE_ACCOUNT options you also have to reuse profiles and do not create a new profile for each printed document.

1.4.11.4.190 SelectGMailAccount2

The SelectGMailAccount2 starts a GMail OAuth authentication process to select the GMail account for sending OAuth email

```c
HRESULT SelectGMailAccount2(
    [in, string] BSTR p_pwsActionId,
    [in] LONG p_nAccountType
);
```

Parameters:
- p_pwsActionId
  - [in] Action id
- p_nAccountType
  - [in] oauth gmail type constant, can be OAUTH_GMAIL_SAME_ACCOUNT or OAUTH_GMAIL_REUSE_ACCOUNT

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong oauth gmail type constant for p_nAccountType
- NV_ERROR_GMAIL - GMail authentication failed

Remarks:
This function will select the GMail account for current loaded profile and provided action id. The account will be saved for the profile only when calling SaveProfile.
GMail OAuth authentication starts the default Internet browser where you can enter the GMail account and authenticate.

There are three modes that can be used with GMail OAuth email type:
- OAUTH_GMAIL_SAME_ACCOUNT: all emails will be sent with the same account, for all users that use this profile. The credentials are saved with the profile and reused for emails. The GMail account can be selected from Profile Manger or calling SelectGMailAccount. Credentials are not asked when the email is sent.
- OAUTH_GMAIL_ASK_ACCOUNT: GMail credentials are not saved with the profile. Each time an email is sent the user will be asked to select the GMail account and authenticate.
- OAUTH_GMAIL_REUSE_ACCOUNT: GMail credentials are saved with the profile, but they are different for each user that uses the profile. User will be asked to select account and authenticate when the first email is sent then the credentials are reused for the same user. The GMail Account can also be selected or changed from the Save As dialog - Recipients dialog or calling SelectGMailAccount.
Take care that if you wish to reuse GMail account credentials with one of the OAUTH_GMAILSAME_ACCOUNT or OAUTH_GMAIL_REUSE_ACCOUNT options you also have to reuse profiles and do not create a new profile for each printed document.

1.4.11.4.191 SetActionOptionBool

The **SetActionOptionBool** method sets an option of boolean type for an action

```c
HRESULT SetActionOptionBool(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [in] BOOL    p_bValue
);
```

**Parameters:**
- **p_pwsActionId**
  - [in] Action id
- **p_wsOption**
  - [in] option constant
- **p_bValue**
  - [in] the value of the option to set.

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.192 SetActionOptionBool2

The **SetActionOptionBool2** method sets an option of boolean type for an action

```c
HRESULT SetActionOptionBool2(
    [in, string] BSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [in] BOOL    p_bValue
);
```

**Parameters:**
- **p_pwsActionId**
  - [in] Action id
- **p_wsOption**
  - [in] option constant
- **p_bValue**
  - [in] the value of the option to set.
Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_ACTION - wrong action id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.193 SetActionOptionEncryptedString

The SetActionOptionEncryptedString method sets an option of encrypted string type for an action

HRESULT SetActionOptionEncryptedString(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [in, string] LPCWSTR p_wsValue
);

Parameters:

p_pwsActionId  
[in] Action id

p_wsOption  
[in] option constant

p_wsValue  
[in] the value of the option to set.

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_ACTION - wrong action id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.194 SetActionOptionEncryptedString2

The SetActionOptionEncryptedString2 method sets an option of encrypted string type for an action

HRESULT SetActionOptionEncryptedString2(
SetActionOption

```c
HRESULT SetActionOption(
    [in, string] BSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [in, string] BSTR    p_wsValue
);
```

**Parameters:**
- `p_pwsActionId` [in] Action id
- `p_wsOption` [in] option constant
- `p_wsValue` [in] the value of the option to set.

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.195 SetActionOptionFloat

The `SetActionOptionFloat` method sets an option of float type for an action.

```c
HRESULT SetActionOptionFloat(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [in] FLOAT   p_fValue
);
```

**Parameters:**
- `p_pwsActionId` [in] Action id
- `p_wsOption` [in] option constant
- `p_fValue` [in] the value of the option to set.

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

### 1.4.11.4.196 SetActionOptionFloat2

The **SetActionOptionFloat2** method sets an option of float type for an action

```c
HRESULT SetActionOptionFloat2(
    [in, string] BSTR p_pwsActionId,
    [in] LONG p_nOption,
    [in] FLOAT p_fValue
);
```

**Parameters:**

- `p_pwsActionId`: [in] Action id
- `p_wsOption`: [in] option constant
- `p_fValue`: [in] the value of the option to set.

**Return values:**

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

### 1.4.11.4.197 SetActionOptionLong

The **SetActionOptionLong** method sets an option of float type for an action

```c
HRESULT SetActionOptionLong(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG p_nOption,
    [in] LONG p_nValue
);
```

**Parameters:**

- `p_pwsActionId`: [in] Action id
- `p_wsOption`: [in] option constant
- `p_nValue`: [in] the value of the option to set.
SetActionOptionLong2
The SetActionOptionLong2 method sets an option of float type for an action

```c
HRESULT SetActionOptionLong2(
    [in, string] BSTR p_pwsActionId,
    [in] LONG    p_nOption,
    [in] LONG    p_nValue
);
```

Parameters:
- `p_pwsActionId` [in] Action id
- `p_wsOption` [in] option constant
- `p_nValue` [in] the value of the option to set.

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
1.4.11.4.199 SetActionOptionString

The **SetActionOptionString** method sets an option of string type for an action

```c
HRESULT SetActionOptionString(
    [in, string] LPCWSTR p_pwsActionId,
    [in] LONG p_nOption,
    [in, string] LPCWSTR p_wsValue
);
```

**Parameters:**

- `p_pwsActionId` [in] Action id
- `p_wsOption` [in] option constant
- `p_wsValue` [in] the value of the option to set.

**Return values:**

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.200 SetActionOptionString2

The **SetActionOptionString2** method sets an option of string type for an action

```c
HRESULT SetActionOptionString2(
    [in, string] BSTR p_pwsActionId,
    [in] LONG p_nOption,
    [in, string] BSTR p_wsValue
);
```

**Parameters:**

- `p_pwsActionId` [in] Action id
- `p_wsOption` [in] option constant
- `p_wsValue` [in] the value of the option to set.

**Return values:**

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_ACTION - wrong action id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.201 SetActiveProfile

The **SetActiveProfile** sets the active profile (i.e. the profile that will be used for printing).

```
HRESULT SetActiveProfile(
    [in] LPCWSTR p_wsProfileId);
```

Parameters:
- p_wsProfileId
  - [in] pointer to a null terminated Unicode string that contains the id of the profile that is to be set as active

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - the profile specified by p_wsProfileId does not exist
- NV_PUBLIC_PROFILE - active profile cannot be change due to propagate active profile flag
- NV_PROFILE_ERROR - private active profile cannot be set because private profiles are not allowed

1.4.11.4.202 SetActiveProfile2

The **SetActiveProfile2** sets the active profile (i.e. the profile that will be used for printing).

```
HRESULT SetActiveProfile2(
    [in] BSTR* p_wstrProfileId);
```

Parameters:
- p_wstrProfileName
  - [in] pointer to a BSTR that contains the id of the profile that is to be set as active

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - the profile specified by p_wstrProfileId does not exist
- NV_PUBLIC_PROFILE - active profile cannot be change due to propagate active profile flag
- NV_PROFILE_ERROR - private active profile cannot be set because private profiles are not allowed

1.4.11.4.203 SetCustomProperty

The **SetCustomProperty** method changes the custom property for the given index
HRESULT SetCustomProperty(
    [in] LONG p_nIndex,
    [in, string] LPCWSTR p_wsPropertyName,
    [in, string] LPCWSTR p_wsPropertyValue,
    [in] BOOL p_bEnabled
);

Parameters:
    p_nIndex
        [in] custom property index
    p_wsPropertyName
        [in] custom property name
    p_wsPropertyValue
        [in] custom property value
    p_bEnabled
        [in] flag, custom property is enabled

Return values:
    S_OK on success or COM error code
    NV_NOT_INITIALIZED - Initialize was not called
    NV_UNKNOWN_PROFILE - no profile loaded
    NV_INVALID_CUSTOM_PROPERTY - invalid custom property index

Remarks:
This method changes information for the custom property at position given by p_nIndex

1.4.11.4.204 SetCustomProperty2

Enter topic text here.

1.4.11.4.205 SetDefaultPrinter

The SetDefaultPrinter method sets the current printer (the one specified in Initialize) as default printer.

HRESULT SetDefaultPrinter(void);

Return values:
    S_OK on success or COM error code
    NV_NOT_INITIALIZED - Initialize was not called

Remarks:
After calling SetDefaultPrinter with an INovaPdfOptions object, call RestoreDefaultPrinter with the same object to restore the original default printer. Do not call SetDefaultPrinter twice, without calling RestoreDefaultPrinter between the calls or else the original default printer will not be restored.

1.4.11.4.206 SetFontOption

The SetFontOption method sets embed options for a given font

HRESULT SetFontOption(
    [in, string] LPCWSTR p_wsFontName,
The `SetFontOption2` method sets embed options for a given font.

```c
HRESULT SetFontOption2(
    [in, string] BSTR p_wsFontName,
    [in] BOOL p_bAlwaysEmbed,
    [in] BOOL p_bNeverEmbed
);
```

**Parameters:**
- `p_wsFontName`: [in] font name
- `p_bAlwaysEmbed`: [out] always embed flag for the font
- `p_bNeverEmbed`: [out] never embed flag for the font

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong font name

### 1.4.11.4.208 SetLayoutOptionBool

The `SetLayoutOptionBool` method sets an option of boolean type for a layout object.

```c
HRESULT SetLayoutOptionBool(
    [in, string] LPCWSTR p_pwsObjectId,
    [in, string] LPCWSTR p_pwsLayoutId,
    [in] LONG p_nOption,
    [in] BOOL p_bValue
);
```

**Parameters:**
- `p_pwsObjectId`: [in] layout object ID
- `p_pwsLayoutId`: [in] layout ID
- `p_nOption`: [in] option number
- `p_bValue`: [in] boolean value

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong font name
Parameters:

- **p_pwsObjectId**
  
  [in] object id (watermark text, watermark image, overlay, signature or content)

- **p_pwsLayoutId**
  
  [in] layout id (obtained with GetLayout)

- **p_wsOption**
  
  [in] option constant

- **p_bValue**
  
  [in] the value of the option to set.

Return values:

- **S_OK** on success or COM error code
- **NV_NOT_INITIALIZED** - Initialize was not called
- **NV_UNKNOWN_PROFILE** - no profile loaded
- **NV_INVALID_OPTION** - wrong option constant
- **NV_PROFILE_ERROR** - cannot find option in profile
- **NV_WRONG_OPTION_TYPE** - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

For layout options, see Working with Layout objects.

### 1.4.11.4.209 SetLayoutOptionBool2

The `SetLayoutOptionBool2` method sets an option of boolean type for a layout object

```c
HRESULT SetLayoutOptionBool2(
    [in, string] BSTR p_pwsObjectId,
    [in, string] BSTR p_pwsLayoutId,
    [in] LONG p_nOption,
    [in] BOOL p_bValue);
```

Parameters:

- **p_pwsObjectId**
  
  [in] object id (watermark text, watermark image, overlay, signature or content)

- **p_pwsLayoutId**
  
  [in] layout id (obtained with GetLayout2)

- **p_wsOption**
  
  [in] option constant

- **p_bValue**
  
  [in] the value of the option to set.

Return values:

- **S_OK** on success or COM error code
- **NV_NOT_INITIALIZED** - Initialize was not called
- **NV_UNKNOWN_PROFILE** - no profile loaded
- **NV_INVALID_OPTION** - wrong option constant
- **NV_PROFILE_ERROR** - cannot find option in profile
- **NV_WRONG_OPTION_TYPE** - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include
folder of novaPDF SDK installation folder you can find the definitions for all options in the next files:
novaOptions.h, novaOptions.pas, Globals.bas.
For layout options, see Working with Layout objects.

1.4.11.4.210 SetLayoutOptionFloat

The SetLayoutOptionFloat method sets an option of float type for a layout object

```c
HRESULT SetLayoutOptionFloat(
    [in, string] LPCWSTR p_pwsObjectId,
    [in, string] LPCWSTR p_pwsLayoutId,
    [in] LONG p_nOption,
    [in] FLOAT p_fValue);
```

Parameters:
- `p_pwsObjectId` [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_pwsLayoutId` [in] layout id (obtained with GetLayout)
- `p_nOption` [in] option constant
- `p_fValue` [in] the value of the option to set.

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
For layout options, see Working with Layout objects.

1.4.11.4.211 SetLayoutOptionFloat2

The SetLayoutOptionFloat2 method sets an option of boolean type for a layout object

```c
HRESULT SetLayoutOptionFloat2(
    [in, string] BSTR p_pwsObjectId,
    [in, string] BSTR p_pwsLayoutId,
    [in] LONG p_nOption,
    [in] FLOAT p_fValue);
```

Parameters:
- `p_pwsObjectId` [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_pwsLayoutId` [in] layout id (obtained with GetLayout2)
- `p_nOption` [in] option constant
- `p_fValue`
[in] the value of the option to set.

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

For layout options, see Working with Layout objects.

1.4.11.4.212 SetLayoutOptionLong

The `SetLayoutOptionLong` method sets an option of long type for a layout object

```c
HRESULT SetLayoutOptionLong(
    [in, string] LPCWSTR p_pwsObjectId,
    [in, string] LPCWSTR p_pwsLayoutId,
    [in] LONG p_nOption,
    [in] LONG p_lValue);
```

Parameters:

- `p_pwsObjectId` [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_pwsLayoutId` [in] layout id (obtained with GetLayout )
- `p_wsOption` [in] option constant
- `p_lValue` [in] the value of the option to set.

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

For layout options, see Working with Layout objects.

1.4.11.4.213 SetLayoutOptionLong2

The `SetLayoutOptionLong2` method sets an option of long type for a layout object

```c
HRESULT SetLayoutOptionLong2(
    [in, string] BSTR p_pwsObjectId,
    [in, string] BSTR p_pwsLayoutId,
```
[in] LONG p_nOption,
[in] LONG p_lValue
);

Parameters:

p_pwsObjectId
[in] object id (watermark text, watermark image, overlay, signature or content)
p_pwsLayoutId
[in] layout id (obtained with GetLayout2)
p_wsOption
[in] option constant
p_lValue
[in] the value of the option to set.

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

For layout options, see Working with Layout objects.

1.4.11.4.214 SetLayoutOptionString

The SetLayoutOptionString method sets an option of string type for a layout object

HRESULT SetLayoutOptionString(
    [in, string] LPCWSTR p_pwsObjectId,
    [in, string] LPCWSTR p_pwsLayoutId,
    [in] LONG p_nOption,
    [in] LPCWSTR p_wsValue
);

Parameters:

p_pwsObjectId
[in] object id (watermark text, watermark image, overlay, signature or content)
p_pwsLayoutId
[in] layout id (obtained with GetLayout)
p_wsOption
[in] option constant
p_wsValue
[in] the value of the option to set.

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long
Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
For layout options, see Working with Layout objects.

1.4.11.4.215 SetLayoutOptionString2

The SetLayoutOptionString2 method sets an option of string type for a layout object

```c
HRESULT SetLayoutOptionString2(
    [in, string] BSTR p_pwsObjectId,
    [in, string] BSTR p_pwsLayoutId,
    [in] LONG p_nOption,
    [in] BSTR p_wsValue
);
```

Parameters:
- `p_pwsObjectId` [in] object id (watermark text, watermark image, overlay, signature or content)
- `p_pwsLayoutId` [in] layout id (obtained with GetLayout2)
- `p_wsOption` [in] option constant
- `p_wsValue` [in] the value of the option to set.

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
For layout options, see Working with Layout objects.

1.4.11.4.216 SetOptionBool

The SetOptionBool method sets a printing option of boolean type

```c
HRESULT SetOptionBool(
    [in] LONG    p_nOption,
    [in] BOOL    p_bValue
);
```

Parameters:
- `p_nOption` [in] option constant
- `p_bValue` [in] long integer value to set

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.217 SetOptionEncryptedString
The SetOptionEncryptedString method sets an encrypted string option

HRESULT SetOptionEncryptedString(
    [in] LONG p_nOption,
    [in] LPCWSTR p_wsValue
);

Parameters:
p_wsOption
    [in] option constant
p_wsValue
    [in] the value of the option to set

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.218 SetOptionEncryptedString2
The SetOptionEncryptedString2 method sets an encrypted string option

HRESULT SetOptionEncryptedString2(
    [in] LONG p_nOption,
    [in] BSTR p_wsValue
);

Parameters:
p_wsOption
    [in] option constant
p_wsValue
    [in] the value of the option to set.
Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.219 SetOptionLong

The **SetOptionLong** method sets a printing option of long int type

```c
HRESULT SetOptionLong(
    [in] LONG    p_nOption,
    [in] LONG    p_lValue
);
```

Parameters:
- p_nOption [in] option constant
- p_lValue [in] long integer value to set

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.220 SetOptionString

The **SetOptionString** method sets a printing option of string type

```c
HRESULT SetOptionString(
    [in] LONG    p_nOption,
    [in] LPCWSTR p_wsValue
);
```

Parameters:
- p_wsOption [in] option constant
- p_wsValue [in] pointer to a null terminated Unicode string containing the value of the option to set

Return values:
- S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.221 SetOptionString2

The `SetOptionString2` method sets a printing option of string type

```c
HRESULT SetOptionString2(
[in] LONG p_nOption,
[in] BSTR p_wsValue
);
```

Parameters:
- `p_wsOption` [in] option constant
- `p_wsValue` [in] pointer to a null terminated Unicode string containing the value of the option to set.

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas,Globals.bas.

1.4.11.4.222 SetOverlayOptionBool

The `SetOverlayOptionBool` method sets an overlay option of boolean type

```c
HRESULT SetOverlayOptionBool(
[in, string] LPCWSTR p_pwsOverlayId,
[in] LONG    p_nOption,
[in] BOOL    p_bValue
);
```

Parameters:
- `p_pwsOverlayId` [in] overlay id (obtained with `AddOverlay`)
- `p_nOption` [in] option constant
- `p_bValue` [in] boolean value to set

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OVERLAY - wrong overlay id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.223 SetOverlayOptionBool2

The `SetOverlayOptionBool2` method sets an overlay option of boolean type

```c
HRESULT SetOverlayOptionBool2(
    [in, string] BSTR p_pwsOverlayId,
    [in] LONG    p_nOption,
    [in] BOOL    p_bValue
);
```

Parameters:

- `p_pwsOverlayId` - [in] overlay id (obtained with AddOverlay2)
- `p_nOption` - [in] option constant
- `p_bValue` - [in] boolean value to set

Return values:

S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OVERLAY - wrong overlay id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.224 SetOverlayOptionEncryptedString

The `SetOverlayOptionEncryptedString` method sets an overlay option of encrypted string type

```c
HRESULT SetOverlayOptionString(
    [in, string] LPCWSTR p_pwsOverlayId,
    [in]         LONG    p_nOption,
    [in, string] LPCWSTR p_wsValue
);
```

Parameters:

- `p_pwsOverlayId` - [in] encrypted string

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
The `SetOverlayOptionEncryptedString2` method sets an overlay option of encrypted string type

```c
HRESULT SetOverlayOptionEncryptedString2(
    [in, string] LPWSTR  p_pwsOverlayId,
    [in]         LONG    p_nOption,
    [in, string] LPCWSTR p_wsValue);
```

**Parameters:**
- `p_pwsOverlayId` [in] overlay id (obtained with AddOverlay )
- `p_nOption` [in] option constant
- `p_wsValue` [in] string value to set (will be encrypted)

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OVERLAY - wrong overlay id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

### 1.4.11.4.226 SetOverlayOptionFloat

The `SetOverlayOptionFloat` method sets an overlay option of float type

```c
HRESULT SetOverlayOptionFloat(
    [in, string] LPCWSTR p_pwsOverlayId,
    [in]         FLOAT    p_flOption,
    [in]         LPCWSTR p_wsValue);
```

**Parameters:**
- `p_pwsOverlayId` [in] overlay id (obtained with AddOverlay )
- `p_nOption` [in] option constant
- `p_wsValue` [in] string value to set (will be encrypted)

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OVERLAY - wrong overlay id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
[in] LONG p_nOption,
[in] FLOAT p_fValue
);

Parameters:
  p_pwsOverlayId
  [in] overlay id (obtained with AddOverlay )
  p_nOption
  [in] option constant
  p_fValue
  [in] float value to set

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_UNKNOWN_PROFILE - no profile loaded
  NV_INVALID_OVERLAY - wrong overlay id
  NV_INVALID_OPTION - wrong option constant
  NV_PROFILE_ERROR - cannot find option in profile
  NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas,Globals.bas.

1.4.11.4.227  SetOverlayOptionFloat2

The SetOverlayOptionFloat2 method sets an overlay option of float type

HRESULT SetOverlayOptionFloat2(
  [in, string] BSTR p_pwsOverlayId,
  [in] LONG p_nOption,
  [in] FLOAT p_fValue
);

Parameters:
  p_pwsOverlayId
  [in] overlay id (obtained with AddOverlay2 )
  p_nOption
  [in] option constant
  p_fValue
  [in] float value to set

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called
  NV_UNKNOWN_PROFILE - no profile loaded
  NV_INVALID_OVERLAY - wrong overlay id
  NV_INVALID_OPTION - wrong option constant
  NV_PROFILE_ERROR - cannot find option in profile
  NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
1.4.11.4.228  SetOverlayOptionLong

The `SetOverlayOptionLong` method sets an overlay option of long type

```c
HRESULT SetOverlayOptionLong(
    [in, string] LPCWSTR p_pwsOverlayId,
    [in] LONG    p_nOption,
    [in] LONG    p_lValue
);
```

Parameters:
- `p_pwsOverlayId` [in] overlay id (obtained with AddOverlay)
- `p_nOption` [in] option constant
- `p_lValue` [in] long value to set

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWNPROFILE - no profile loaded
- NV_INVALIDOVERLAY - wrong overlay id
- NV_INVALIDOPTION - wrong option constant
- NV_PROFILEERROR - cannot find option in profile
- NV_WRONGOPTIONTYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.229  SetOverlayOptionLong2

The `SetOverlayOptionLong2` method sets an overlay option of long type

```c
HRESULT SetOverlayOptionLong2(
    [in, string] BSTR p_pwsOverlayId,
    [in] LONG    p_nOption,
    [in] LONG    p_lValue
);
```

Parameters:
- `p_pwsOverlayId` [in] overlay id (obtained with AddOverlay)
- `p_nOption` [in] option constant
- `p_lValue` [in] long value to set

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWNPROFILE - no profile loaded
- NV_INVALIDOVERLAY - wrong overlay id
- NV_INVALIDOPTION - wrong option constant
- NV_PROFILEERROR - cannot find option in profile
- NV_WRONGOPTIONTYPE - option is not of type long
Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.230 SetOverlayOptionString

The `SetOverlayOptionString` method sets an overlay option of string type

```c
HRESULT SetOverlayOptionString(
    [in, string] LPCWSTR p_pwsOverlayId,
    [in]         LONG    p_nOption,
    [in, string] LPCWSTR p_wsValue
);
```

Parameters:
- `p_pwsOverlayId` [in] overlay id (obtained with `AddOverlay`)
- `p_nOption` [in] option constant
- `p_wsValue` [in] string value to set

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - `Initialize` was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_OVERLAY - wrong overlay id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.231 SetOverlayOptionString2

The `SetOverlayOptionString2` method sets an overlay option of string type

```c
HRESULT SetOverlayOptionString2(
    [in, string] BSTR p_pwsOverlayId,
    [in]         LONG    p_nOption,
    [in, string] BSTR p_wsValue
);
```

Parameters:
- `p_pwsOverlayId` [in] overlay id (obtained with `AddOverlay2`)
- `p_nOption` [in] option constant
- `p_wsValue` [in] string value to set

Return values:
- S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_OVERLAY - wrong overlay id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.232 SetOverrideOptionBool
The SetOverrideOptionBool method sets an override printing option of boolean type

```c
HRESULT SetOverrideOptionBool(
    [in] LONG    p_nOption,
    [in] BOOL    p_bValue
);
```

Parameters:
- p_nOption
  [in] option constant
- p_bValue
  [in] bool value to set

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - wrong option constant

Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.233 SetOverrideOptionEncryptedString
The SetOverrideOptionEncryptedString method sets an override printing option of encrypted string type

```c
HRESULT SetOverrideOptionEncryptedString(
    [in] LONG    p_nOption,
    [in, string] LPCWSTR p_wsValue
);
```

Parameters:
- p_nOption
  [in] option constant
- p_wsValue
  [in] string value to set

Return values:
- S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_INVALID_OPTION - wrong option constant

Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.234 SetOverrideOptionEncryptedString2

The **SetOverrideOptionEncryptedString2** method sets an override printing option of encrypted string type.

```c
HRESULT SetOverrideOptionEncryptedString2(
    [in] LONG    p_nOption,
    [in, string] BSTR p_wsValue
);
```

Parameters:

- `p_nOption` [in] option constant
- `p_wsValue` [in] string value to set

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - wrong option constant

Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.235 SetOverrideOptionLong

The **SetOverrideOptionLong** method sets an override printing option of long type.

```c
HRESULT SetOverrideOptionLong(
    [in] LONG    p_nOption,
    [in] BOOL    p_bValue
);
```

Parameters:

- `p_nOption` [in] option constant
- `p_bValue` [in] long integer value to set

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - wrong option constant

Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
Override options are not part of the profile. These options override the options from the active profile.
Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.236 SetOverrideOptionString

The **SetOverrideOptionString** method sets an override printing option of string type

```c
HRESULT SetOverrideOptionString(
    [in] LONG    p_nOption,
    [in, string] LPCWSTR p_wsValue
);
```

**Parameters:**
- `p_nOption` [in] option constant
- `p_wsValue` [in] string value to set

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - wrong option constant

Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.237 SetOverrideOptionString2

The **SetOverrideOptionString2** method sets an override printing option of string type

```c
HRESULT SetOverrideOptionString2(
    [in] LONG    p_nOption,
    [in, string] BSTR p_wsValue
);
```

**Parameters:**
- `p_nOption` [in] option constant
- `p_wsValue` [in] string value to set

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_INVALID_OPTION - wrong option constant

Remarks:
You can find the complete list of override options Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

Override options are not part of the profile. These options override the options from the active profile.

1.4.11.4.238 SetPrinterActivePublicProfile

The SetPrinterActivePublicProfile method sets an active public profile for a printer

```
HRESULT SetPrinterActivePublicProfile(
    [in, string] LPCWSTR p_wsPrinterName,
    [in, string] LPCWSTR p_wsProfileId
);
```

**Parameters:**
- `p_wsPrinterName` [in, string] printer name
- `p_wsProfileId` [in, string] public profile id

**Return values:**
- S_OK on success or COM error code
- NV_SERVICE_ERROR - error connecting to novaPDF Server service

**Remarks:**
Sets an public profile as the active profile for the printer. All users will be forced to use this public profile, the cannot change the active profile.

1.4.11.4.239 SetPrinterActivePublicProfile2

The SetPrinterActivePublicProfile2 method sets an active public profile for a printer

```
HRESULT SetPrinterActivePublicProfile2(
    [in, string] BSTR p_wsPrinterName,
    [in, string] BSTR p_wsProfileId
);
```

**Parameters:**
- `p_wsPrinterName` [in, string] printer name
- `p_wsProfileId` [in, string] public profile id

**Return values:**
- S_OK on success or COM error code
- NV_SERVICE_ERROR - error connecting to novaPDF Server service

**Remarks:**
Sets an public profile as the active profile for the printer. All users will be forced to use this public profile, the cannot change the active profile.

1.4.11.4.240 SetPrinterOption

The SetPrinterOption method sets a general printer option of long int type

```
HRESULT SetPrinterOption(
    [in] LONG p_nOption,
```


[in] LONG p_lValue

Parameters:
  p_nOption [in] option constant
  p_lValue [in] long integer value to set

Return values:
  S_OK on success or COM error code
  NV_NOT_INITIALIZED - Initialize was not called, printer name unknown
  NV_INVALID_OPTION - cannot set a general (all users) option on client computers

Remarks:
Sets general options that are not part of a profile (like show or not the select profile dialog. You can find the complete list of option names in the Profile option strings chapter, general settings tables. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.241 SetPrinterPublicProfile

The SetPrinterPublicProfile method sets the visibility of a public profile for a printer

HRESULT SetPrinterPublicProfile(
    [in, string] LPCWSTR p_wsPrinterName,
    [in, string] LPCWSTR p_wsProfileId,
    [in] BOOL p_bVisible
);

Parameters:
  p_wsPrinterName [in, string] printer name
  p_wsProfileId [in, string] public profile id
  p_bVisible [in] flag, if visible

Return values:
  S_OK on success or COM error code
  NV_SERVICE_ERROR - error connecting to novaPDF Server service

Remarks:
Sets the visibility option for a public profile and a printer. By default, all public profiles are visible ton all printers. They can be hidden for some printers using this method.

1.4.11.4.242 SetPrinterPublicProfile2

The SetPrinterPublicProfile2 method sets the visibility of a public profile for a printer

HRESULT SetPrinterPublicProfile2(
    [in, string] BSTR p_wsPrinterName,
    [in, string] BSTR p_wsProfileId,
    [in] BOOL p_bVisible
);


Parameters:

- `p_wsPrinterName` [in, string] printer name
- `p_wsProfileId` [in, string] public profile id
- `p_bVisible` [in] flag, if visible

Return values:

- S_OK on success or COM error code
- NV_SERVICE_ERROR - error connecting to novaPDF Server service

Remarks:

Sets the visibility option for a public profile and a printer. By default, all public profiles are visible to all printers. They can be hidden for some printers using this method.

### 1.4.11.4.243 SetPrinterServerFlags

The `SetPrinterServerFlags` method sets the profiles usage options for a printer.

```c
HRESULT SetPrinterServerFlags(
    [in, string] LPCWSTR p_wsPrinterName,
    [in] BOOL p_bAllowPrivateProfiles,
    [in] BOOL p_bShowSelectProfile,
    [in] BOOL p_bAllowHideDialog
);
```

Parameters:

- `p_wsPrinterName` [in, string] printer name
- `p_bAllowPrivateProfiles` [in] allow private profiles
- `p_bShowSelectProfile` [in] show select profiles dialog for all users
- `p_bAllowHideDialog` [in] allow users to hide select profiles dialog

Return values:

- S_OK on success or COM error code
- NV_SERVICE_ERROR - error connecting to novaPDF Server service

### 1.4.11.4.244 SetPrinterServerFlags2

The `SetPrinterServerFlags2` method sets the profiles usage options for a printer.

```c
HRESULT SetPrinterServerFlags2(
    [in, string] BSTR p_wsPrinterName,
    [in] BOOL p_bAllowPrivateProfiles,
    [in] BOOL p_bShowSelectProfile,
    [in] BOOL p_bAllowHideDialog
);
```

Parameters:
p_wsPrinterName
  [in, string] printer name
p_bAllowPrivateProfiles
  [in] allow private profiles
p_bShowSelectProfile
  [in] show select profiles dialog for all users
p_bAllowHideDialog
  [in] allow users to hide select profiles dialog

Return values:
S_OK on success or COM error code
NV_SERVICE_ERROR - error connecting to novaPDF Server service

1.4.11.4.245 SetSaveRule

The SetSaveRule method sets values for the save rule at the given index

HRESULT SetSaveRule(
    [in] LONG p_nIndex,
    [in] LONG p_nType,
    [in, string] LPCWSTR p_wsName,
    [in, string] LPCWSTR p_wsDescription,
    [in, string] LPCWSTR p_wsText,
    [in, string] LPCWSTR p_wsFormat,
    [in] BOOL p_bEnabled
);

Parameters:

p_nIndex
  [in] save rule index
p_nType
  [in] save rule type (0-remove from start, 1- remove from end, 2 - remove all, 3 - regular expression)
p_wsName
  [in] rule name
p_wsDescription
  [in] rule description
p_wsText
  [in] text to remove or regular expression
p_wsFormat
  [in] format for regular expression
p_bEnabled
  [in] flag, tag is enabled

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_SAVE_RULE - invalid save rule index

Remarks:
This method sets information for the save rule at given index
1.4.11.4.246  SetSaveRule2

The **SetSaveRule2** method sets values for the save rule at the given index

```c
HRESULT SetSaveRule2(
    [in] LONG p_nIndex,
    [in] LONG p_nType,
    [in, string] BSTR p_wsName,
    [in, string] BSTR p_wsDescription,
    [in, string] BSTR p_wsText,
    [in, string] BSTR p_wsFormat,
    [in] BOOL p_bEnabled
);}
```

**Parameters:**
- p_nIndex: [in] save rule index
- p_nType: [in] save rule type (0-remove from start, 1- remove from end, 2 - remove all, 3 - regular expression)
- p_wsName: [in] rule name
- p_wsDescription: [in] rule description
- p_wsText: [in] text to remove or regular expression
- p_wsFormat: [in] format for regular expression
- p_bEnabled: [in] flag, tag is enabled

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_SAVE_RULE - invalid save rule index

**Remarks:**
This method sets information for the save rule at given index

1.4.11.4.247  SetUserTag

The **SetUserTag** method changes the user tag for the given index

```c
HRESULT SetUserTag(
    [in] LONG p_nIndex,
    [in, string] LPCWSTR p_wsTag,
    [in, string] LPCWSTR p_wsDefaultValue,
    [in] BOOL p_bEnabled
);}
```

**Parameters:**
- p_nIndex: [in] save rule index
- p_wsTag: [in] user tag name
- p_wsDefaultValue: [in] default value
- p_bEnabled: [in] flag, tag is enabled
The `SetUserTag2` method changes the user tag for the given index.

```c
HRESULT SetUserTag2(
    [in] LONG p_nIndex,
    [in, string] BSTR p_wsTag,
    [in, string] BSTR p_wsDefaultValue,
    [in] BOOL p_bEnabled
);
```

**Parameters:**
- `p_nIndex` [in] tag index
- `p_wsTag` [in] user tag
- `p_wsDefaultValue` [in] default tag value
- `p_bEnabled` [in] flag, tag is enabled

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_USER_TAG - invalid user tag index

**Remarks:**
This method changes information for the tag at position given by `p_nIndex`. 

1.4.11.4.248 SetUserTag2

The `SetUserTag2` method changes the user tag for the given index.
1.4.11.4.249 SetWatermarkImageOptionBool

The `SetWatermarkImageOptionBool` method sets a watermark image option of boolean type

```c
HRESULT SetWatermarkImageOptionBool(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in] BOOL    p_bValue
);
```

**Parameters:**
- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkImage)
- `p_nOption` [in] option constant
- `p_bValue` [in] boolean value to set

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.250 SetWatermarkImageOptionBool2

The `SetWatermarkImageOptionBool2` method sets a watermark image option of boolean type

```c
HRESULT SetWatermarkImageOptionBool2(
    [in, string] BSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in] BOOL    p_bValue
);
```

**Parameters:**
- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkImage2)
- `p_nOption` [in] option constant
- `p_bValue` [in] boolean value to set

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long
Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.251 SetWatermarkImageOptionEncryptedString

The **SetWatermarkImageOptionEncryptedString** method sets a watermark image option of encrypted string type

```c
HRESULT SetWatermarkImageOptionEncryptedString(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in] LONG p_nOption,
    [in, string] LPCWSTR p_wsValue);
```

Parameters:
- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkImage)
- `p_wsOption` [in] option constant
- `p_wsValue` [in] the value of the option to set.

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.252 SetWatermarkImageOptionEncryptedString2

The **SetWatermarkImageOptionEncryptedString2** method sets a watermark image option of encrypted string type

```c
HRESULT SetWatermarkImageOptionEncryptedString2(
    [in, string] BSTR p_pwsWatermarkId,
    [in] LONG p_nOption,
    [in, string] BSTR p_wsValue);
```

Parameters:
- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkImage2)
- `p_wsOption` [in] option constant
- `p_wsValue`
[in] the value of the option to set.

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.253 SetWatermarkImageOptionFloat

The `SetWatermarkImageOptionFloat` method sets a watermark image option of float type

```c
HRESULT SetWatermarkImageOptionFloat(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in] FLOAT   p_fValue);
```

Parameters:

- `p_fValue` [in] float value to set

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.254 SetWatermarkImageOptionFloat2

The `SetWatermarkImageOptionFloat2` method sets a watermark image option of float type

```c
HRESULT SetWatermarkImageOptionFloat2(
    [in, string] BSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in] FLOAT   p_fValue);
```

Parameters:

- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkImage )
- `p_nOption` [in] option constant
- `p_fValue` [in] float value to set

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
Parameters:

- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkImage2)
- `p_nOption` [in] option constant
- `p_fValue` [in] float value to set

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.255  SetWatermarkImageOptionLong

The `SetWatermarkImageOptionLong` method sets a watermark image option of long int type

```c
HRESULT SetWatermarkImageOptionLong(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in] LONG    p_lValue
);
```

Parameters:

- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkImage)
- `p_nOption` [in] option constant
- `p_lValue` [in] long integer value to set

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
1.4.11.4.256 SetWatermarkImageOptionLong2

The `SetWatermarkImageOptionLong2` method sets a watermark image option of long int type.

```c
HRESULT SetWatermarkImageOptionLong2(
    [in, string] BSTR p_pwsWatermarkId,
    [in]    LONG    p_nOption,
    [in]    LONG    p_lValue
);
```

**Parameters:**
- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkImage2)
- `p_nOption` [in] option constant
- `p_lValue` [in] long integer value to set

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.257 SetWatermarkImageOptionString

The `SetWatermarkImageOptionString` method sets a watermark image option of string type.

```c
HRESULT SetWatermarkImageOptionString(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in]    LONG    p_nOption,
    [in, string] LPCWSTR p_wsValue
);
```

**Parameters:**
- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkImage)
- `p_wsOption` [in] option constant
- `p_wsValue` [in] the value of the option to set.

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long
Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.258 SetWatermarkImageOptionString2

The `SetWatermarkImageOptionString2` method sets a watermark image option of string type

```
HRESULT SetWatermarkImageOptionString2(
    [in, string] BSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in, string] BSTR    p_wsValue
);
```

Parameters:
- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkImage2 )
- `p_wsOption` [in] option constant
- `p_wsValue` [in] the value of the option to set.

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_IMG - wrong watermark id
- NVINVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.259 SetWatermarkTextOptionBool

The `SetWatermarkTextOptionBool` method sets a watermark text option of boolean type

```
HRESULT SetWatermarkTextOptionBool(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in] BOOL    p_bValue
);
```

Parameters:
- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkText )
- `p_nOption` [in] option constant
- `p_bValue` [in] boolean value to set

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
NV_UNKNOWN_PROFILE - no profile loaded
NV_INVALID_WATERMARK_TXT - wrong watermark id
NV_INVALID_OPTION - wrong option constant
NV_PROFILE_ERROR - cannot find option in profile
NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.260 SetWatermarkTextOptionBool2

The SetWatermarkTextOptionBool2 method sets a watermark text option of boolean type

```c
HRESULT SetWatermarkTextOptionBool2(
    [in, string] BSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in] BOOL    p_bValue
);
```

Parameters:
- p_pwsWatermarkId
  - [in] watermark id (obtained with AddWatermarkText2)
- p_nOption
  - [in] option constant
- p_bValue
  - [in] boolean value to set

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_TXT - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.261 SetWatermarkTextOptionFloat

The SetWatermarkTextOptionFloat method sets a watermark text option of float type

```c
HRESULT SetWatermarkTextOptionFloat(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in] FLOAT   p_fValue
);
```

Parameters:
- p_pwsWatermarkId
  - [in] watermark id (obtained with AddWatermarkText )
- p_nOption
  - [in] option constant
SetWatermarkTextOptionFloat2

The `SetWatermarkTextOptionFloat2` method sets a watermark text option of float type

```
HRESULT SetWatermarkTextOptionFloat2(
    [in, string] BSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in] FLOAT   p_fValue);
```

Parameters:
- `p_pwsWatermarkId`: [in] watermark id (obtained with AddWatermarkText2)
- `p_nOption`: [in] option constant
- `p_fValue`: [in] float value to set

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_TXT - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

SetWatermarkTextOptionLong

The `SetWatermarkTextOptionLong` method sets a watermark text option of long int type

```
HRESULT SetWatermarkTextOptionLong(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in] LONG    p_lValue);
```

Parameters:
- `p_pwsWatermarkId`: [in] watermark id (obtained with AddWatermarkText2)
- `p_nOption`: [in] option constant
- `p_lValue`: [in] long value to set

Return values:
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_TXT - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
Parameters:

- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkText )
- `p_nOption` [in] option constant
- `p_lValue` [in] long integer value to set

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_TXT - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.264 SetWatermarkTextOptionLong2

The SetWatermarkTextOptionLong2 method sets a watermark text option of long int type

```
HRESULT SetWatermarkTextOptionLong2(
    [in, string] BSTR p_pwsWatermarkId,
    [in] LONG p_nOption,
    [in] LONG p_lValue
);
```

Parameters:

- `p_pwsWatermarkId` [in] watermark id (obtained with AddWatermarkText2 )
- `p_nOption` [in] option constant
- `p_lValue` [in] long integer value to set

Return values:

- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_TXT - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

Remarks:

You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.
1.4.11.4.265 SetWatermarkTextOptionString

The `SetWatermarkTextOptionString` method sets a watermark text option of string type

```c
HRESULT SetWatermarkTextOptionString(
    [in, string] LPCWSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in, string] LPCWSTR p_wsValue
);
```

**Parameters:**
- `p_pwsWatermarkId`: [in] watermark id (obtained with AddWatermarkText)
- `p_wsOption`: [in] option constant
- `p_wsValue`: [in] the value of the option to set.

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_TXT - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long

**Remarks:**
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.266 SetWatermarkTextOptionString2

The `SetWatermarkTextOptionString2` method sets a watermark text option of string type

```c
HRESULT SetWatermarkTextOptionString2(
    [in, string] BSTR p_pwsWatermarkId,
    [in] LONG    p_nOption,
    [in, string] BSTR    p_wsValue
);
```

**Parameters:**
- `p_pwsWatermarkId`: [in] watermark id (obtained with AddWatermarkText2)
- `p_wsOption`: [in] option constant
- `p_wsValue`: [in] the value of the option to set.

**Return values:**
- S_OK on success or COM error code
- NV_NOT_INITIALIZED - Initialize was not called
- NV_UNKNOWN_PROFILE - no profile loaded
- NV_INVALID_WATERMARK_TXT - wrong watermark id
- NV_INVALID_OPTION - wrong option constant
- NV_PROFILE_ERROR - cannot find option in profile
- NV_WRONG_OPTION_TYPE - option is not of type long
Remarks:
You can find the complete list of option names in the Profile option strings chapter. In the Include folder of novaPDF SDK installation folder you can find the definitions for all options in the next files: novaOptions.h, novaOptions.pas, Globals.bas.

1.4.11.4.267 UnRegisterEventWindow

The UnRegisterEventWindow unregisters the window registered with RegisterEventWindow so it will no longer receive printing messages.

HRESULT UnRegisterEventWindow(void);

Return values:
S_OK on success or COM error code
NV_NOT_INITIALIZED - Initialize was not called

1.4.11.4.268 WaitForNovaEvent

The WaitForNovaEvent waits for a Windows event that will be signaled by the printer

HRESULT WaitForNovaEvent(
    [in] LONG p_nMilliseconds
    [out] BOOL* p_bTimeout
);

Parameters:
    p_nMilliseconds
        [in] Number of milliseconds to wait for the event. See How to use events for more details.
p_bTimeout
        [out] returns TRUE if a the wait function return with a timeout, and FALSE otherwise

Return values:
S_OK - on success
S_FALSE - event was not found

1.5 Samples

1.5.1 What sample to choose

There are several modes to start a print job to novaPDF SDK 11, and depending on your application, you should choose a different sample:

1. If you a C++ application and you create a printer job using Windows API calls like OpenPrinter, StartDoc,... you should check the Hello World sample.

2. If you have a Delphi application and you print using the Printer object provided by Delphi, check the Hello World Delphi sample.

3. If you have a C# application that prints using the package "System.Drawing.Printing", check the Hello World CSharp sample.

4. If you have a VBNet application that prints using the package "System.Drawing.Printing", check the Hello World VBNet sample.
5. If you have an ASP.NET application that prints using the package "System.Drawing.Printing", check the Hello World ASPNET sample.

6. If you have a Java application that prints using the package "System.Drawing.Printing", check the Hello World Java sample.

7. If you have an Access database and you want to generate PDF files, check the PDF Reports Access sample.

8. If you perform a print job by calling other controls "Print()" method, or if you print an existing document using "ShellExecute()" function, you should check one of these samples: C++ MFC Converter, Delphi VCLConverter, CSharp Converter, VBNet Converter.

9. If your application runs on a network check the C++ Hello World (network) sample.

10. If you have a C++ document/view MFC architecture check the MFC Scribble sample.

11. If you want to convert MS Word documents or if you use other OLE controls to print your documents, choose one of the next samples: Word OLE CSharp, Word OLE Delphi, Word OLE VBNet or Word OLE (Java).

12. If you wish to convert Microsoft Office documents (Word, Excel, PowerPoint, Publisher or Visio), choose one of the next samples: Convert Office Docs C++, Convert Office Docs C#, Convert Office Docs VBNet, Convert Office Docs Delphi.

13. If you wish to work with temporary printers check the C++ Temporary printer sample.

14. If you wish to use novaPDF SDK in a multithreading application check the C++ Multiple printers sample.

15. If you do not wish to work with option profiles but only change a few options like pdf file name check one of next samples: Override Options C++, Override Options CSharp, Override Options Delphi, Override Options VBNet.

All "Hello World" samples include a separate tool file with sample functions for setting all kind of options (save options, watermarks, bookmarks, overlay, graphics, document info, security, email, links, fonts).

1.5.2 Access

1.5.2.1 PDF Reports

PDF Reports Sample is an Access database with one table and one form. On the form you can set several options for the novaPDF SDK 11 and then press a button to generate a PDF file. A report is made on the table and it is sent to novaPDF SDK 11.

It demonstrates the basic use of the INovaPDFOptions interface. The printing job is made using the object "Application.Printer"
Basically the sample creates a new profile called "Access Profile", sets the new profile as active, sets the user options from form controls, opens and prints a report, and restores original printer settings.

**Source code**

```vba
Option Compare Database
Option Explicit

Private objPDF As Object

Const strIAProfile As String = "Access Profile"
Const strPDFDriver As String = "novaPDF SDK 11"
Const bIAPublicProfile As Long = 0

Private Sub cmdCreatePDF_Click()
    On Error GoTo Error_cmdCreatePDF_Click

    Dim strActiveProfileId As String
    Dim strNewProfileId As String
    Dim strDefaultPrinter As String

    ' create the NovaPdfOptions object
    Dim objPDF As New NovaPdfOptions

    ' initialize the NovaPdfOptions object to use with a printer licensed for SDK
    objPDF.Initialize strPDFDriver ""

    ' Store the Default Printer. * Note - Access cannot use the objPDF.SetDefaultPrinter
    ' update Access internally fast enough. You must use the Application.Printer.
    strDefaultPrinter = Application.Printer.DeviceName

    ' Get the Active Profile
    objPDF.GetActiveProfile2 strActiveProfileId

    ' Add new profile
    objPDF.AddProfile2 strProfileName, bIAPublicProfile, strNewProfileId

    ' Load the profile
    objPDF.LoadProfile2 strNewProfileId

With Me

    '****************************

    ' Set save options
    If !optSaveOptions Then
        objPDF.SetOptionLong NOVPDF_SAVE_PROMPT_TYPE, PROMPT_SAVE_EXTENDED
    Else
        objPDF.SetOptionLong NOVPDF_SAVE_PROMPT_TYPE, PROMPT_SAVE_NONE
    End If

    objPDF.SetOptionLong NOVPDF_SAVE_FOLDER_TYPE, SAVEFOLDER_CUSTOM
    objPDF.SetOptionString2 NOVPDF_SAVE_FOLDER, txtSaveFolder
    objPDF.SetOptionString2 NOVPDF_SAVE_FILE_NAME, !txtFilename
    objPDF.SetOptionLong NOVPDF_SAVE_FILEEXIST_ACTION, !cboWhenFileExists

    ' After Save Action
    objPDF.SetOptionBool NOVPDF_ACTION_DEFAULT_VIEWER, !chkOpenViewer

    ' .... other options
```

```
' save options
objPDF.SaveProfile

' Set the PDF print driver.
objPDF.SetActiveProfile2 strNewProfileId

' Set the Default printer.
Set Application.Printer = Application.Printers(strPDFDriver)

' Run the selected report and create a PDF file.
DoCmd.OpenReport "rptSMZipCode"

' Return to previous settings
objPDF.SetActiveProfile2 strActiveProfileId

objPDF.DeleteProfile2 strNewProfileId

' Restore the Default Printer.
Set Application.Printer = Application.Printers(strDefaultPrinter)
End With

Exit_cmdCreatePDF_Click:
    Set objPDF = Nothing
Exit Sub

Error_cmdCreatePDF_Click:
    Debug.Print Err.Number & ":" & Err.Description
    Resume Next
End Sub

1.5.3 ASP.NET

1.5.3.1 Hello World

Hello World (ASP.NET) sample is a simple ASP application that generates one PDF file containing the text "novaPDF says Hello World from ASP.NET". The PDF is created using the novaPDF SDK 11 printer driver and is saved in the “upload” folder. It demonstrates the basic use of the INovaPDFOptions interface. The printing job is done using the package System.Drawing.Printing

What this sample does:
- determines the active profile, makes a copy of it and names it "Test ASP.NET"
- sets the new profile (Test ASP.NET) as active as well as some mandatory settings
- generates a test PDF file and saves it in the “upload” folder
- restores the original settings of the novaPDF SDK 11 printer driver.

Note
To test this sample on IIS, you should set the Application Pool to run under the “Local System” Account. Here’s how you can do this:
1. In IIS Manager, expand Local computer, the Application Pools folder, right-click the application pool you would like to configure (the one selected for this application/ virtual directory) and click Properties.
2. Go to the Identity tab.
3. Click on Predefined and in the list box beside it click Local System. Click OK.
**SOURCE CODE FOR THE ASP SAMPLE OUTPUT DISPLAY PAGE (DEFAULT.ASPX):**

```html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head id="Head1" runat="server">
    <title>Hello World ASP.NET</title>
</head>
<body>
<h1>Hello World ASP.NET and novaPDF SDK</h1>
<form id="form2" runat="server">
    <asp:Label ID="Label1" runat="server" Text="Press the button"></asp:Label>
    <br />
    <asp:LinkButton ID="Link1" Visible="false" runat="server" View folder."/asp:LinkButton>
    <br />
    <asp:Button ID="Button1" runat="server" onclick="Button1_Click" Text="Print to novaPDF" />
    <br />
    <br />
    1. In IIS Manager, expand the local computer, expand the Application Pools folder, right-click the application pool you would like to configure (the one selected for this application/ virtual directory), and click Properties.
    <br />
    2. Click the Identity tab.
    <br />
    3. Click Predefined, and in the list box beside it, click Local System.
    <br />
    4. Click OK.
</form>
</body>
</html>
```

**SOURCE CODE FOR THE ASP SAMPLE (DEFAULT.ASPX.CS):**

```csharp
using System;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Drawing;
using System.Drawing.Printing;

// the novapiLib package must be added as a COM reference
using novapiLib;

public partial class _Default : System.Web.UI.Page
{
    protected void Button1_Click(object sender, EventArgs e)
    {
    }
}
try
{
    // create the NovaPdfOptions object
    NovaPdfOptions11 pNova = new NovaPdfOptions11();
    // initialize the NovaPdfOptions object
    pNova.InitializeSilent(PRINTER_NAME, "");
    // get the active profile ...
    string activeProfile = null;
    string newProfileId = null;
    pNova.SetActiveProfile(out activeProfile);
    // create new profile
    pNova.AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC, out newProfileId);
    pNova.LoadProfile(newProfileId);
    // and set some options
    pNova.SetOptionString2(NovaOptions.NOVAPDF_DOCINFO_SUBJECT, "ASP.NET Hello document");
    pNova.SetOptionString(NovaOptions.NOVAPDF_SAVE_FILE_NAME, novaPDFDocument);
    pNova.SetOptionLong(NovaOptions.NOVAPDF_SAVE_FOLDER_TYPE, (int)SaveFolder.SAVEFOLDER_CUSTOM);
    pNova.SetOptionString(NovaOptions.NOVAPDF_SAVE_FOLDER, Server.MapPath("upload"));
    pNova.SetOptionLong(NovaOptions.NOVAPDF_SAVE_FILEEXIST_ACTION, (int)SaveFileConflictType.FILE_CONFLICT_STRATEGY_AUTONUMBER_NEW);
    pNova.SetOptionBool(NovaOptions.NOVAPDF_INFO_VIEWER_ENABLE, 1);
    pNova.SetOptionLong(NovaOptions.NOVAPDF_SAVE_PROMPT_TYPE, (int)SaveDlgType.PROMPT_SAVE_SIMPLE);

    // save the new added profile
    pNova.SaveProfile();
    // set the new profile as the active profile
    pNova.SetActiveProfile(newProfileId);

    // print a test page, using the previously set active profile settings
    using (PrintDocument pd = new PrintDocument())
    {
        pd.PrintController = new StandardPrintController();
        pd.PrinterSettings.PrinterName = PRINTER_NAME;
        pd.PrintPage += new PrintPageEventHandler(PrintPageFunction);
        pd.Print();
    }

    if ((activeProfile.Length > 0) && (activeProfile.CompareTo(PROFILE_NAME) != 0))
    {
        pNova.SetActiveProfile(activeProfile);
    }
    pNova.DeleteProfile(newProfileId);
    // mark finish changing options so they are saved for the printer
Label1.Text = "Success.<br />You will find the new printed file in "/upload" folder.";
Button1.Visible = false;

Link1.Visible = true;
Link1.Attributes.Add("href", "upload/");
Link1.Attributes.Add("target", "_blank");
}
catch (System.Runtime.InteropServices.COMException come)
{
    Label1.Style.Add("color", "red");
    Label1.Text = "COM Exception:" + come.Message;
}
catch (Exception ee)
{
    Label1.Style.Add("color", "red");
    Label1.Text = "Exception:" + ee.Message;
    return;
}

// and finally the function that actually prints the page
private static void PrintPageFunction(object sender,
PrintPageEventArgs ev)
{
    string str = "novaPDF says Hello World from ASP.NET";
    Font font = new Font("Arial", 16);
    Brush brush = new SolidBrush(Color.Black);
    ev.Graphics.DrawString(str, font, brush, 20.0f, 20.0f);
    ev.HasMorePages = false;
}

1.5.4 Delphi
1.5.4.1 VCL Converter

The VCL Converter sample demonstrates how to convert an existing file by printing it to novaPDF SDK 11 using the ShellExecute function. It also demonstrates how to set different options and manage profiles.

The same approach should be taken if you print using a "Print()" method from another object (like an internet browser or a report control). Just replace the ShellExecute call with the call of your Print method.

When the application starts, it creates a few profiles and makes different settings in the profiles. Then it shows a dialog from where the user can select the active profile and change its settings using the controls from the dialog.

After that a document can be selected from the harddisk and printed to novaPDF SDK 11 using the ShellExecute function call.

When using this technique to convert a file to PDF, you have to take care of the fact that ShellExecute prints to the default printer. This function returns immediately and does not wait until the print is
finished (it may return before the printing is actually started). Therefore you have to set the default printer to novaPDF SDK 11 before calling ShellExecute (using the SetDefaultPrinter method), register FileSaved message (or any other novaPDF SDK 11 message) to be sure that the print job was started. In this message handler restore the default printer (with the RestoreDefaultPrinter method). This way you made sure that the default printer was restored and your document is printed to novaPDF SDK 11.

Source code snippets

1. DECLARE INovaPdfOptions variable

```pascal
declare an INovaPdfOptions member variable
PRIVATE
m_novaOptions : INovaPdfOptions;
```

2. Register novaPDF SDK 11 messages

```pascal
register event messages
WM_NOVAPDF2_FILESAVED := RegisterWindowMessage(MSG_NOVAPDF2_FILESAVED);
WM_NOVAPDF2_PRINTERROR := RegisterWindowMessage(MSG_NOVAPDF2_PRINTERROR);

// handle event messages
PUBLIC
PROCEDURE WndProc(var Message: TMessage); override;
PROCEDURE TForm1.WndProc(var Message: TMessage);
BEGIN
  IF Message.Msg = WM_NOVAPDF2_FILESAVED then BEGIN
    // ...
  END ELSE IF Message.Msg = WM_NOVAPDF2_PRINTERROR then BEGIN
    // ...
  END ELSE BEGIN
    inherited WndProc(Message);
  END;
END;
```

3. Initialize INovaPdfOptions

```pascal
PROCEDURE TForm1.FormCreate(Sender: TObject);
BEGIN
  // ...
  // initialize COM libraries
  hr := ActiveX.CoInitialize(nil);
  if FAILED(hr) then begin
    MessageDlg('Failed to initialize COM' + #13+SysErrorMessage(hr) + #13+.SysErrorMessage(GetLastError()), mtWarning, [mbOK], 0);
  end;

  // create an instance of INovaPdfOptions
  m_novaOptions := nil;
  hr := ActiveX.CoCreateInstance(CLASS_NovaPdfOptions11,
  // CLSID_CNovaPdfSource,
  nil,
  CLSCTX_INPROC_SERVER,
  IID_INovaPdfOptions11,
  m_novaOptions);
  if (FAILED(hr)) then begin
    MessageDlg('Failed to create novaPDF COM object', mtWarning, [mbOK], 0
```
// initialize the NovaPdfOptions object to use with a printer licensed for SDK
m_novaOptions.Initialize(PRINTER_NAME, '');
if (FAILED(hr)) then begin
    MessageDlg('Failed to initialize NovaPdfOptions', mtWarning, [mbOK], 0);
    exit;
end;

// add 2 profiles
CreateProfiles();

// load profiles in list
LoadProfiles();

4. Release INovaPDFOptions
PROCEDURE TForm1.FormClose(Sender: TObject; var Action: TCloseAction); BEGIN
    //...
    // delete profiles
    m_novaOptions.DeleteProfile2(m_strSmallSizeProfileID);
    m_novaOptions.DeleteProfile2(m_strFullOptProfileID);
    // destroy m_novaOptions object
    // - no need for this as the Delphi takes care of it automatically
    // uninitialize COM libraries
    ActiveX.CoUninitialize();
    //...
END;

5. Set novaPDF SDK 11 Options
PROCEDURE TForm1.CreateProfiles(); BEGIN
    // Add a profile called "Small size"
    m_novaOptions.AddProfile2(SMALL_SIZE_PROFILE, PROFILE_IS_PUBLIC,
    m_strSmallSizeProfileID);
    m_novaOptions.LoadProfile2(m_strSmallSizeProfileID);
    // Set some options to this profile
    // disable the "Save PDF file as" prompt
    m_novaOptions.SetOptionLong(NOVPDF_SAVE_PROMPT_TYPE, PROMPT_SAVE_NONE);
    // set generated Pdf files destination folder to the application path
    m_novaOptions.SetOptionLong(NOVPDF_SAVE_FOLDER_TYPE, SAVEFOLDER_CUSTOM);
    m_novaOptions.SetOptionString2(NOVPDF_SAVE_FOLDER,
    ExtractFilePath(Application.ExeName));
    // set output file name
    m_novaOptions.SetOptionString2(NOVPDF_SAVE_FILE_NAME, 'PDF Converter small size.pdf');
6. Start a print job
PROCEDURE TForm1.btnStartPrintClick(Sender: TObject);
var
  hExec : HINST;
BEGIN
  //...  
  // set the active profile to be used for printing  
  m_novaOptions.SetActiveProfile2(strProfileId);
  // register our window to receive messages from the printer  
  m_novaOptions.RegisterEventWindow(self.Handle);
  // set novaPDF as default printer, so it will be used by ShellExecute  
  m_novaOptions.SetDefaultPrinter();
  // license the file to be converted with ShellExecute  
  m_novaOptions.LicenseShellExecuteFile(efFileToConvert.Text);
  // print the document  
  m_bPrintJobPending := TRUE;
  hExec := ShellAPI.ShellExecute(self.handle, 'print', PChar(efFileToConvert.Text), PChar(''), PChar(''), SW_HIDE);
  if (hExec <= 32) then begin // failed to execute program
    m_bPrintJobPending := FALSE;
    m_novaOptions.UnRegisterEventWindow();
    m_novaOptions.RestoreDefaultPrinter();
  end;
END;

7. Restore default printer when printing finished
PROCEDURE TForm1.WndProc(var Message: TMessage);
BEGIN
  if Message.Msg = WM_NOVAPDF2_FILESAVED then begin
    // restore original default printer  
    m_novaOptions.UnRegisterEventWindow();
    m_novaOptions.RestoreDefaultPrinter();
    m_bPrintJobPending := FALSE;
  end else if Message.Msg = WM_NOVAPDF2_PRINTERROR then begin
    case (Message.WParam) of
      ERROR_MSG_TEMP_FILE : begin
        MessageDlg('Error saving temporary file on printer server', mtWarning, [mbOK], 0);
        end;
      ERROR_MSG_LIC_INFO : begin
        end;
      end;
END;
MessageDlg('Error reading license information', mtWarning, [mbOK], 0);
end;
ERROR_MSG_SAVE_PDF : begin
  MessageDlg('Error saving PDF file', mtWarning, [mbOK], 0);
end;
ERROR_MSG_JOB_CANCELED : begin
  MessageDlg('Print job was canceled', mtWarning, [mbOK], 0);
end;
// restore original default printer
m_novaOptions.UnRegisterEventWindow();
m_novaOptions.RestoreDefaultPrinter();
m_bPrintJobPending := FALSE;
end else begin
  inherited WndProc(Message);
end;
END;

1.5.4.2 Hello World Delphi

Hello World Delphi sample is a simple Windows console application that prints one page with the "Hello World from Delphi!" text to the novaPDF SDK 11.

It demonstrates the basic use of the INovaPDFOptions interface. The printing job is made with calls to the global Printer object defined by Delphi. Text is printed using Canvas.TextOut method.

It generates a "Hello World.pdf" file in the working folder.

Notice

If you print an existing document using "ShellExecute()" function or you want to handle printing events, you should check the VCL Converter sample instead.

Source code

program HelloWorld;

{$APPTYPE CONSOLE}

uses
  ActiveX,
  Printers,
  Windows,
  novaOptions in '..\..\..\include\novaOptions.pas',
  novapiLIB80_TLB in '..\..\..\include\novapiLIB11_TLB.pas',
  Nova in 'Nova.pas';

const

  //name of novaPDF Printer
  PRINTER_NAME = 'novaPDF SDK 11';
//text to be written in the PDF file
PDF_TEXT = 'Hello world from Delphi!';

//PDF file name
PDF_FILE_NAME = 'HelloWorld_Delphi';

//Print profile name
PROFILE_NAME = 'HelloWorld Delphi Profile';
PROFILE_IS_PUBLIC = 0;

var
hr : HRESULT;
pNova : INovaPdfOptions11;
strOldActiveProfileID : WideString;
strNewProfileID : WideString;
//decomment next code if you use workaround for printer index (see below)
//Device, Driver, Port: array[0..80] of Char;
//DevMode: THandle;
begin
  //initialize COM
  hr := ActiveX.CoInitialize(nil);
  if (FAILED (hr)) then begin
    System.Writeln('Failed to initialize COM');
    exit;
  end;

  //create one NovaPdfOptions instance
  pNova := nil;
  hr := ActiveX.CoCreateInstance(CLASS_NovaPdfOptions11,
      //CLSID_CNovaPdfSource,
      nil,
      CLSCTX_INPROC_SERVER,
      IID_INovaPdfOptions11,
      pNova);
  if (FAILED(hr)) then begin
    System.Writeln('Failed to create novaPDF COM object');
    exit;
  end;

  // initialize the NovaPdfOptions object to use with a printer licensed for SDK
  pNova.Initialize2( PRINTER_NAME, '');
  pNova.SetDefaultPrinter();

  // now the default printer is novaPDF printer but the Printer object is not updated
  // here is a workaround to update the Printer object with the default printer
  // you only need this code if you check later on the Printer.PrinterIndex
to find out the default printer
  //Printer.GetPrinter(Device, Driver, Port, DevMode);
  //Printer.SetPrinter(PRINTER_NAME, Driver, Port, 0);

  //create a new profile with default settings
  pNova.AddProfile2(PROFILE_NAME, PROFILE_IS_PUBLIC, strNewProfileID);
//load the newly created profile
if SUCCEEDED(hr) and (Length(strNewProfileID) > 0) then begin
    pNova.LoadProfile2(strNewProfileID);
end else begin
    System.Writeln('Failed to create profile');
    exit;
end;

// set novaPDF options
// uncomment the function calls for the options you wish to set and
// change the options in nova.cpp unit
//AddProfileOptions(pNova);
//AddDocumentInformation(pNova);
//AddViewerOptions(pNova);
//AddLinksOptions(pNova);
//AddAdvancedOptions(pNova);
//AddGraphicsOptions(pNova);
//AddEmbedFontsOptions(pNova);
//AddBookmarksDefinitions(pNova);
//AddSecurity(pNova);
//AddSaveOptions(pNova);
//AddAfterSaveActions(pNova);
//AddEmailOptions(pNova);
//AddWatermarkImage(pNova);
//AddWatermarkText(pNova);
//AddPageContentOptions(pNova);
//AddOverlayOptions(pNova);
//AddSignatureOptions(pNova);

//save profile changes
pNova.SaveProfile();

//get current default profile id
pNova.GetActiveProfile2(strOldActiveProfileID);
//set as active profile for printer
pNova.SetActiveProfile2(strNewProfileID);

//start print job
Printer.Title := PDF_FILE_NAME;
Printer.BeginDoc();
Printer.Canvas.Font.Size := 24;
Printer.Canvas.TextOut( 100, 80, PDF_TEXT);
Printer.endDoc();
System.Writeln('Print job finished');

//restore default profile
pNova.SetActiveProfile2(strOldActiveProfileID);
pNova.DeleteProfile2(strNewProfileID);
//resore default printer
pNova.RestoreDefaultPrinter();

//release NovaPdfOptions
// pNova._Release();

ActiveX.CoUninitialize();

end.
1.5.4.3 Convert Office Docs

The **Convert Office Docs** sample is a simple Windows console application that converts some Microsoft Office documents (Word, Excel, Publisher, PowerPoint and Visio) using Convert... functions. It uses Office OLE automation so Microsoft Office has to be installed.

The sample documents converted are located in the "OfficeDocuments" folder in novaPDF SDK installation folder. The pdf files are saved in the same folder.

Hidden links and internal document links are converted in pdf hyperlinks (except for Excel documents).

**Source code**

```delphi
program ConvertOfficeDocs;

{$APPTYPE CONSOLE}

uses
  ActiveX,
  Printers,
  Windows,
  sysutils,
  novaOptions in '..\..\..\include\novaOptions.pas',
  novaEvents in '..\..\..\include\novaEvents.pas',
  novapiLIB10_TLB in '..\..\..\include\novapiLIB11_TLB.pas',
  program ConvertOfficeDocs;

const

// name of novaPDF Printer
PRINTER_NAME    = 'novaPDF SDK 11';

// PDF file name
PDF_FILE_NAME   = 'ConvertOffice_Delphi';

// Print profile name
PROFILE_NAME    = 'Convert Office Delphi';
PROFILE_IS_PUBLIC = 0;

var
  hr : HRESULT;
  pNova : INovaPdfOptions10;
  strOldActiveProfileID : WideString;
  strNewProfileID : WideString;
  bTimeout : Integer;
  strDocumentsPath : WideString;
  strDocPath : WideString;
  strHeadings : WideString;

begin

  // initialize COM
  hr := ActiveX.CoInitialize(nil);
  if (FAILED (hr)) then begin
    System.Writeln('Failed to initialize COM');
    exit;
  end;
```
//create one NovaPdfOptions instance
pNova := nil;
hr := ActiveX.CoCreateInstance(CLASS_NovaPdfOptions11,
    nil,
    CLSCTX_INPROC_SERVER,
    IID_INovaPdfOptions11,
    pNova);

if (FAILED(hr)) then begin
    System.Writeln('Failed to create novaPDF COM object');
    exit;
end;

// initialize the NovaPdfOptions object to use with a printer licensed for SDK
// if you have an application license for novaPDF SDK,
// pass the license key to the Initialize() function
pNova.Initialize2(PRINTER_NAME, '');

//create a new profile with default settings
pNova.AddProfile2(PROFILE_NAME, PROFILE_IS_PUBLIC, strNewProfileID);

//load the newly created profile
if SUCCEEDED(hr) and (Length(strNewProfileID) > 0) then begin
    pNova.LoadProfile2(strNewProfileID);
end else begin
    System.Writeln('Failed to create profile');
    exit;
end;

//DO NOT OPEN PDF
//an Open action is added by default in the profile - delete or disable this open
//pNova.DisableActionType(NOVA_ACTION_OPEN);

// set novaPDF save options
AddSaveOptions(pNova);

//save profile changes
pNova.SaveProfile();

try
    //get current default profile id
    pNova.GetActiveProfile2(strOldActiveProfileID);
except
    strOldActiveProfileID := '';
end;

//set as active profile for printer
pNova.SetActiveProfile2(strNewProfileID);

//Path for novaPDF 10 SDK sample documents
strDocumentsPath := GetEnvironmentVariable('ALLUSERSPROFILE');
strDocumentsPath := strDocumentsPath + '\Documents
ovaPDF 10\SDK\OfficeDocuments'
strDocPath := strDocumentsPath;
strDocPath := strDocPath + 'Word.docx';
//style name | level | type
strHeadings := 'Heading 1|1|0;Heading 2|2|0;Heading 3|3|0';
pNova.LicenseApplication('WINWORD.EXE');
pNova.LicenseApplication('SPLWOW64.EXE');
pNova.ConvertWordDocument2(strDocPath, 1, 0, 1, 1, 1, 1, 1, 1, 3, strHeadings);

// Convert PowerPoint document
// Microsoft Office PowerPoint has to be installed
// --------------------------------------------------
//Document file path
strDocPath := strDocumentsPath;
strDocPath := strDocPath + 'PowerPoint.pptx';
pNova.LicenseApplication('POWERPNT.EXE');
pNova.LicenseApplication('SPLWOW64.EXE');
pNova.ConvertPowerPointDocument2(strDocPath, 1, 1, 1, 1, 1, 1);

// --------------------------------------------------
// Convert Publisher document
// Microsoft Office Publisher has to be installed
// --------------------------------------------------
//Document file path
strDocPath := strDocumentsPath;
strDocPath := strDocPath + 'Publisher.pub';
pNova.LicenseApplication('MSPUB.EXE');
pNova.LicenseApplication('SPLWOW64.EXE');
pNova.ConvertPublisherDocument2(strDocPath, 1, 1, 1, 1, 1, 1);

// --------------------------------------------------
// Convert Excel document
// Microsoft Office Excel has to be installed
// hidden links in Excel documents will not be converted to pdf links
// --------------------------------------------------
//Document file path
strDocPath := strDocumentsPath;
strDocPath := strDocPath + 'Excel.xlsx';
pNova.LicenseApplication('EXCEL.EXE');
pNova.LicenseApplication('SPLWOW64.EXE');
pNova.ConvertExcelDocument2(strDocPath, 1);

// --------------------------------------------------
// Convert Visio document
// Microsoft Office Visio has to be installed
// --------------------------------------------------
//Document file path
strDocPath := strDocumentsPath;
strDocPath := strDocPath + 'Visio.vsd';
pNova.LicenseApplication('VISIO.EXE');
pNova.LicenseApplication('SPLWOW64.EXE');
pNova.ConvertVisioDocument2(strDocPath, 1, 1, 1, 1, 1);

if Length(strOldActiveProfileID) > 0 then begin
  //restore default profile
  pNova.SetActiveProfile2(strOldActiveProfileID);
end;
pNova.DeleteProfile2(strNewProfileID);
ActiveX.CoUninitialize();
end.

1.5.4.4 Word OLE Delphi

The Word OLE Delphi sample is a simple Windows console application that converts a MS Word document (C:\Test.doc) to PDF using Word OLE automation.

Source code

program WordOLEDelphi;

{$APPTYPE CONSOLE}

uses
  ActiveX,
  Printers,
  ComObj,
  SysUtils,
  Dialogs,
  novaOptions in '..\..\..\include\novaOptions.pas',
  novapiLIB80_TLB in '..\..\..\include\novapiLIB11_TLB.pas';

const
  //name of novaPDF Printer
  PRINTER_NAME = 'novaPDF SDK 11';

  //Print profile name
  PROFILE_NAME = 'Test OLE Delphi Profile';
  PROFILE_IS_PUBLIC = 0;

var
  hr : HRESULT;
  pNova : I novaPdfOptions11;
  strOldActiveProfileID : WideString;
  strNewProfileID : WideString;
  Word : VARIANT;
  NewDoc : VARIANT;

begin

  //initialize COM
  hr := ActiveX.CoInitialize(nil);
  if (FAILED (hr)) then begin
    System.Writeln('Failed to initialize COM');
    exit;
  end;

  //create one NovaPdfOptions instance
  pNova := nil;
  hr := ActiveX.CoCreateInstance(CLASS_NovaPdfOptions11,
      //CLSID_CNovaPdfSource,
      nil,
      CLSCTX_INPROC_SERVER,
      IID_INovaPdfOptions11,
pNova); if (FAILED(hr)) then begin
  System.Writeln('Failed to create novaPDF COM object');
  exit;
end;

//initialize NovaPdfOptions and pass printer name
pNova.Initialize2(PRINTER_NAME, '');
pNova.SetDefaultPrinter();

// now the default printer is novaPDF printer but the Printer object is
not updated
// here is a workaround to update the Printer object with the default
printer
// you only need this code if you check later on the Printer.PrinterIndex
to find out the default printer
//Printer.GetPrinter(Device, Driver, Port, DevMode);
//Printer.SetPrinter(PRINTER_NAME, Driver, Port, 0);

//create a new profile with default settings
pNova.AddProfile2(PROFILE_NAME, PROFILE_IS_PUBLIC, strNewProfileID);

//load the newly created profile
if (Length(strNewProfileID) > 0) then begin
  pNova.LoadProfile2(strNewProfileID);
end else begin
  System.Writeln('Failed to create profile');
  exit;
end;

// set PDF document Title
pNova.SetOptionString2(NOVAOFDOCINFO_TITLE, 'Hello World Delphi Sample');
// set resulting file name
pNova.SetOptionString2(NOVAOFSAVE_FOLDER, 'C:\');
// do not show prompt dialog
pNova.SetOptionLong(NOVAOFSAVE_PROMPT_TYPE, PROMPT_SAVE_NONE);
// if file exists, override
pNova.SetOptionLong(NOVAOFSAVE_FILEEXIST_ACTION, FILE_CONFLICT_STRATEGY_OVERWRITE);
// do not open document in PDF viewer
pNova.SetOptionBool(NOVAOFACTION_DEFAULT_VIEWER, 1);

// save profile changes
pNova.SaveProfile();

// get current default profile id
pNova.GetActiveProfile2(strOldActiveProfileID);
// set as active profile for printer
pNova.SetActiveProfile2(strNewProfileID);

// Print Word Document
try
  pNova.InitializeOLEUsage('Word.Application');
  Word := CreateOleObject('Word.Application');
  Word.DisplayAlerts := 0;
  pNova.LicenseOLEServer();
NewDoc:= Word.Documents.Open('C:\temp\Test.doc', False, True);  
NewDoc.PrintOut(False);  
NewDoc.Close(False);  
Word.Quit(False); 
except 
  on E: Exception do 
    ShowMessage(E.Message); 
end;

//restore default profile 
PNova.SetActiveProfile2(strOldActiveProfileID); 
PNova.DeleteProfile2(strNewProfileID);  
//resore default printer  
PNova.RestoreDefaultPrinter();  

//release NovaPdfOptions  
//pNova._Release();  
ActiveX.CoUninitialize();

end.

1.5.4.5 Override Options

Override Options sample is a simple Windows console application that demonstrates the basic use 
of the INovaPDFOptions interface, without using profiles options. Instead of creating a profile and 
setting options in the profile, the sample uses SetOverrideOptions... functions to quickly set some 
options that will overwrite active profile settings. This is useful if you have a simple application 
where you just wish to change the pdf file name, email address or other simple options for each printed 
document.

There is a limited set of options that can be set with SetOverrideOptions... functions, usually settings 
that users can also change from the Save As dialog or the other prompt dialogs that can be shown 
before printing a document. If you need more complex settings, like adding a watermark, overlay or 
signature you have to configure them in a profile.

You can combine usage of profiles and override options functions. For instance you can create a 
profile with a watermark that is set as active and then use the override function just to change the 
file name for each document.

Once set, override options are taken in account for all printed documents until they are removed 
with DeleteOverrideOptions  function.

The sample prints one page with a simple text to the novaPDF SDK 11 and generates a 
"OverrideOptions.pdf" file in the C:\temp folder.

Source code

program OverrideOptions;

{$APPTYPE CONSOLE}

uses
ActiveX,
Printers,
Windows,
novaOptions in '..\..\..\include\novaOptions.pas',
novaEvents in '..\..\..\include\novaEvents.pas',
novapliLIB10_TLB in '..\..\..\include\novapliLIB11_TLB.pas';

const

//name of novaPDF Printer
PRINTER_NAME = 'novaPDF SDK 11';

//text to be written in the PDF file
PDF_TEXT = 'Hello world from Delphi!';

//PDF file name
PDF_FILE_NAME = 'HelloWorld_Delphi';

var
hr : HRESULT;
pNova : INovaPdfOptions11;
bTimeout : Integer;

begin

//initialize COM
hr := ActiveX.CoInitialize(nil);
if (FAILED (hr)) then begin
  System.Writeln('Failed to initialize COM');
  exit;
end;

//create one NovaPdfOptions instance
pNova := nil;
h := ActiveX.CoCreateInstance(CLASS_NovaPdfOptions11, //CLSID_CNovaPdfSource,
                               nil,
                               CLSCTX_INPROC_SERVER,
                               IID_INovaPdfOptions11,
                               pNova);

if (FAILED(hr)) then begin
  System.Writeln('Failed to create novaPDF COM object');
  exit;
end;

// initialize the NovaPdfOptions object to use with a printer licensed for SDK
// if you have an application license for novaPDF SDK,
// pass the license key to the Initialize() function
pNova.Initialize2(PRINTER_NAME, '');
pNova.SetDefaultPrinter();
// now the default printer is novaPDF printer but the Printer object is not updated
// here is a workaround to update the Printer object with the default printer
// you only need this code if you check later on the Printer.PrinterIndex to find out the default printer
//Printer.GetPrinter(Device, Driver, Port, DevMode);
//Printer.SetPrinter(PRINTER_NAME, Driver, Port, 0);

//Override options from active profile

//Save options
//do not show prompt save dialog
pNova.SetOverrideOptionLong(NOVAPDF_OVERRIDE_SAVE_TYPE, PROMPT_SAVE_NONE);
//save the pdf in next folder
pNova.SetOverrideOptionLong(NOVAPDF_OVERRIDE_SAVE_FOLDER_TYPE, SAVEFOLDER_CUSTOM);
pNova.SetOverrideOptionString2(NOVAPDF_OVERRIDE_SAVE_FOLDER, 'C:\temp');
pNova.SetOverrideOptionString2(NOVAPDF_OVERRIDE_SAVE_FILE, PDF_FILE_NAME);
//access folder, if restricted
pNova.SetOverrideOptionEncryptedString2(NOVAPDF_OVERRIDE_SAVE_USER, 'user');
pNova.SetOverrideOptionEncryptedString2(NOVAPDF_OVERRIDE_SAVE_PASSWORD, 'password ');
//if a file with the same name already exists
pNova.SetOverrideOptionLong(NOVAPDF_OVERRIDE_SAVE_CONFLICT, FILE_CONFLICT_STRATEGY_OVERWRITE);

//disable open PDF in viewer
//pNova.SetOverrideOptionBool(NOVAPDF_OVERRIDE_OPEN_VIEWER, 0);

//Embed fonts
pNova.SetOverrideOptionBool(NOVAPDF_OVERRIDE_EMBEDFONTS, 1);

//Set compression level
//pNova.SetOverrideOptionLong(NOVAPDF_OVERRIDE_COMPRESSION, COMPRESSION_HIGHQUALITY);

//Merge PDF
//Enable merge
{
pNova.SetOverrideOptionBool(NOVAPDF_OVERRIDE MERGE PDF, 1);
//Merge with the same file, if the file exists in destination OR merge with another file
pNova.SetOverrideOptionLong(NOVAPDF OVERRIDE MERGE FILE TYPE, MERGE_FILE_OTHER);
//Append to existing PDF, insert at the beginning or insert at a page number
pNova.SetOverrideOptionLong(NOVAPDF OVERRIDE MERGE TYPE, MERGE_TYPE_APPEND);
//if merge with other file, specify file name (and path)
pNova.SetOverrideOptionString(NOVAPDF OVERRIDE MERGE FILE, 'invoice-text.pdf');
//if merge type is set to insert at page number
//pNova.SetOverrideOptionLong(NOVAPDF OVERRIDE INSERT PAGENO, 1);
//if the existing PDF is encrypted
pNova.SetOverrideOptionEncryptedString2(NOVAPDF OVERRIDE PDF PASSWORD, 'password ');
}
//Document info
pNova.SetOverrideOptionBool(NOVAPDF_OVERRIDE_DOC_INFO, 1);
pNova.SetOverrideOptionString2(NOVAPDF_OVERRIDE_DOC_TITLE, 'sdk title');
pNova.SetOverrideOptionString2(NOVAPDF_OVERRIDE_DOC_SUBJECT, 'sdk subject');
pNova.SetOverrideOptionString2(NOVAPDF_OVERRIDE_DOC_AUTHOR, 'sdk author');
pNova.SetOverrideOptionString2(NOVAPDF_OVERRIDE_DOC_KEY, 'sdk key');
pNova.SetOverrideOptionString2(NOVAPDF_OVERRIDE_DOC_CREATOR, 'sdk creator');

//Encrypt PDF
{
pNova.SetOverrideOptionBool(NOVAPDF_OVERRIDE_SECURITY, 1);
pNova.SetOverrideOptionEncryptedString2(NOVAPDF_OVERRIDE_SECURITY_U, 'user');
pNova.SetOverrideOptionEncryptedString2(NOVAPDF_OVERRIDE_SECURITY_O, 'owner');
}

//Email
pNova.SetOverrideOptionBool(NOVAPDF_OVERRIDE_SEND_EMAIL, 1);
pNova.SetOverrideOptionString2(NOVAPDF_OVERRIDE_REC_TO, 'to');
pNova.SetOverrideOptionString2(NOVAPDF_OVERRIDE_REC_FROM, 'from');
pNova.SetOverrideOptionString2(NOVAPDF_OVERRIDE_REC_CC, 'cc');
pNova.SetOverrideOptionString2(NOVAPDF_OVERRIDE_REC_BCC, 'bcc');

pNova.RegisterNovaEvent2(NOVAPDF_EVENT_ACTIONS);

//start print job
Printer.Title := PDF_FILE_NAME;
Printer.BeginDoc();
Printer.Canvas.Font.Size := 24;
Printer.Canvas.TextOut(100, 80, PDF_TEXT);
Printer.endDoc();
System.Writeln('Print job finished');

pNova.WaitForNovaEvent(120000, bTimeout);

//restore default printer
pNova.RestoreDefaultPrinter();

//release NovaPdfOptions
// pNova._Release();
ActiveXObject.CoUninitialize();

decl

1.5.5 C#

1.5.5.1 Hello World CSharp

Hello World CSharp sample is a simple Windows console application that prints one page with the "novaPDF says Hello World from C#" text to the novaPDF SDK 11.

It demonstrates the basic use of the INovaPDFOptions interface. The printing job is made using the package "System.Drawing.Printing"
Basically the sample determines the active profile, makes a copy of it into a profile called "Test C#", sets the new profile as active, sets the subject of the generated PDF document, prints a page, and restores original printer settings. The location of the generated document depends on whatever the settings are for the current active profile.

**Notice**

Because of the specific exception based error handling in .NET, all calls to methods in the INovaPDFOptions interface must be nested within a try-catch block. Consider for example that we want to add a profile called "test", but the profile "test" already exists. Then the call pNova.AddProfile("test") will throw an "System.Runtime.InteropServices.COMException" with the ErrorCode property set to NV_PROFILE_EXISTS (0x5DA0006).

**Source code**

```csharp
using System;
using System.Drawing;
using System.Drawing.Printing;
using System.Windows.Forms;

namespace Hello_World_CSharp
{
    /// <summary>
    /// Summary description for Class1.
    /// </summary>
    public class Class1
    {
        // The main entry point for the application.
        [STAThread]
        static void Main(string[] args)
        {
            try
            {
                // create the NovaPdfOptions object
                NovaPdfOptions11 pNova = new NovaPdfOptions11();
                // initialize the NovaPdfOptions object
                pNova.Initialize(PRINTER_NAME, "");

                // get the active profile ...
                string activeProfile = null;
                bool isActiveProfile = true;
                //pNova.getActiveProfile(out activeProfile, out nActivePublic);

                try
                {
```
pNova.SetActiveProfile(out activeProfile);
}
catch (System.Runtime.InteropServices.COMException e)
{
    isActiveProfile = false;
    // ignore profile exists error
    if (NovaErrors.NV_NO_ACTIVE_PROFILE == (uint e.ErrorCode)
    {
        System.Console.WriteLine("The printer does not have an active profile");
    }
    else
    {
        // more serious error, propagate it
        throw e;
    }
}

string _newProfileID = String.Empty;
// add a new profile
pNova.AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC, out _newProfileID);

// load the new profile
pNova.LoadProfile(_newProfileID);
// nova.NovaTools.AddProfileOptions(pNova);

// and set some options
// uncomment the function calls for the options you wish to set
// change the options in the nova.cs unit
// nova.NovaTools.AddProfileOptions(pNova);
// nova.NovaTools.AddDocumentInformation(pNova);
// nova.NovaTools.AddViewerOptions(pNova);
// nova.NovaTools.AddLinksOptions(pNova);
// nova.NovaTools.AddAdvancedOptions(pNova);
// nova.NovaTools.AddGraphicsOptions(pNova);
// nova.NovaTools.AddSecurityOptions(pNova);
// nova.NovaTools.AddSaveOptions(pNova);
// nova.NovaTools.AddAfterSaveActions(pNova);
// nova.NovaTools.AddEmailOptions(pNova);
// nova.NovaTools.AddWatermarkImage(pNova);
// nova.NovaTools.AddWatermarkText(pNova);
// nova.NovaTools.AddPageContentOptions(pNova);
// nova.NovaTools.AddOverlayOptions(pNova);
// nova.NovaTools.AddSignatureOptions(pNova);
// nova.NovaTools.AddEmbedFontsOptions(pNova);
// nova.NovaTools.AddBookmarksDefinitions(pNova);

// save the new added profile
pNova.SaveProfile();
// set the new profile as the active profile
pNova.SetActiveProfile(_newProfileID);

// print a test page, using the previously set active profile settings
using (PrintDocument pd = new PrintDocument())
{
    pd.PrinterSettings.PrinterName = PRINTER_NAME;
pd.PrintPage += new PrintPageEventHandler(PrintPageFunction);
pd.Print();

if (isActiveProfile)
{
    pNova.SetActiveProfile(activeProfile);
}
pNova.DeleteProfile(_newProfileID);
}

catch (System.Runtime.InteropServices.COMException e)
{
    MessageBox.Show(e.Message);
}
catch (Exception e)
{
    MessageBox.Show(e.Message);
}

// and finally the function that actually prints the page
private static void PrintPageFunction(object sender,
PrintPageEventArgs ev)
{
    string str = "novaPDF says Hello World from C#";
    Font font = new Font("Arial", 16);
    Brush brush = new SolidBrush(Color.Black);
    ev.Graphics.DrawString(str, font, brush, 20.0f, 20.0f);
    ev.HasMorePages = false;
}

1.5.5.2 CSharp Converter

The CSharp Converter sample demonstrates how to convert an existing file by printing it to novaPDF SDK 11 using the ShellExecute function. It also demonstrates how to set different options and manage profiles.

The same approach should be taken if you print using a "Print()" method from another object (like an internet browser or a report control). Just replace the ShellExecute call with the call of your Print method.

When the application starts, it creates a few profiles and makes different settings in the profiles. Then it shows a dialog from where the user can select the active profile and change its settings using the controls from the dialog.

After that a document can be selected from the harddisk and printed to novaPDF SDK 11 using the ShellExecute function call.

When using this technique to convert a file to PDF, you have to take care of the fact that ShellExecute prints to the default printer. This function returns immediately and does not wait until the print is finished (it may return before the printing is actually started). Therefore you have to set the default
printer to novaPDF SDK 11 before calling ShellExecute (using the SetDefaultPrinter method), wait the process to be started (using WaitForExit()), restore the default printer (with the RestoreDefaultPrinter method). This way you made sure that the default printer was restored and your document is printed to novaPDF SDK 11.

**Source code snippets**

1. **DECLARE INovaPdfOptions variable**

   ```csharp
   private NovaPdfOptions11Class mobjNovaOptios;
   ```

2. **Initialize INovaPdfOptions**

   ```csharp
   mobjNovaOptios = new NovaPdfOptions11Class();
   
   // initialize the NovaPdfOptions object
   mobjNovaOptios.Initialize(PRINTER_NAME, "");
   
   // create option profiles
   AddSmallProfile();
   AddFullProfile();
   ```

3. **Set novaPDF SDK 11 Options**

   ```csharp
   try
   {
   // Set some options to this profile
   mobjNovaOptios.AddProfile(SMALL_SIZE_PROFILE,
   PROFILE_IS_PUBLIC, out newSmallSizeProfileId);
   
   // load the new profile
   mobjNovaOptios.LoadProfile(newSmallSizeProfileId);
   
   // disable the "Save PDF file as" prompt
   mobjNovaOptios.SetOptionLong(NovaOptions
   .NOVAPDF_SAVE_PROMPT_TYPE, 0);
   // set generated Pdf files destination folder   "c:"
   mobjNovaOptios.SetOptionString(NovaOptions.NOVAPDF_SAVE_FOLDER,
   "c:"");
   
   // ......
   }
   catch (System.Runtime.InteropServices.COMException ComException)
   {
   MessageBox.Show("Error creating Small Size Profile:\r\n" +
   ComException.Message);
   System.Diagnostics.Debug.WriteLine(ComException.Message);
   }
   ```

4. **Start a print job**

   ```csharp
   private void btnStartPrinting_Click(object sender, System.EventArgs e)
   {
   UpdateProfileFromDialog();
   ```
if (txtFileToConvert.Text.Trim() == "")
{
    MessageBox.Show("No file to convert!");
    return;
}

mobjNovaOptios.SetActiveProfile((string)(cmbProfiles.SelectedItem.ToString()));
mobjNovaOptios.SetDefaultPrinter();

mobjNovaOptios.LicenseShellExecuteFile(txtFileToConvert.Text);

Process myProcess = new Process();
try
{
    myProcess.StartInfo.FileName = txtFileToConvert.Text;
    myProcess.StartInfo.Verb = "Print";
    myProcess.StartInfo.CreateNoWindow = true;
    myProcess.Start();
}
catch (Win32Exception ex)
{
    if (ex.NativeErrorCode == ERROR_FILE_NOT_FOUND)
    {
        Console.WriteLine(ex.Message + ". Check the path and filename");
    }
    else if (ex.NativeErrorCode == ERROR_ACCESS_DENIED)
    {
        // Note that if your word processor might generate exceptions
        // such as this, which are handled first.
        Console.WriteLine(ex.Message + ". You do not have permission to print this file.");
    }
    myProcess.WaitForExit(10000);
    myProcess.Close();
    mobjNovaOptios.RestoreDefaultPrinter();
}

1.5.5.3 Word OLE CSharp

The Word OLE CSharp sample is a simple Windows console application that converts a MS Word document (C:\Test.doc) to PDF using Word OLE automation.

Source code

using System;
using System.Drawing;
using System.Drawing.Printing;
using System.Windows.Forms;
// the novapiLib package must be added as a COM reference
using novapiLib11;
namespace Hello_World_CSharp
{
    /// <summary>
    /// Summary description for Class1.
    /// </summary>
    class Class1
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
    public static string PRINTER_NAME = "novaPDF SDK 11";
    public static int NOVAPDF_INFO_SUBJECT = 68;
    public static string PROFILE_NAME = "Test C# OLE";
    public static int PROFILE_IS_PUBLIC = 0;
    public static uint NV_PROFILE_EXISTS = 0xD5DA0006;
    public static uint NV_NO_ACTIVE_PROFILE = 0xD5DA0028;

        [STAThread]
    static void Main(string[] args)
    {
        // create the NovaPdfOptions11 object
        NovaPdfOptions11 pNova = new NovaPdfOptions11();
        // get the active profile ...
        string activeProfile = "";
        string _newProfileID = String.Empty;

        try
        {
            // initialize the NovaPdfOptions11 object
            // if you have an application license for novaPDF
            pNova.Initialize(PRINTER_NAME, "");

            try
            {
                pNova.GetActiveProfile(out activeProfile);
            }
            catch (System.Runtime.InteropServices.COMException e)
            {
                // ignore profile exists error
                if (NV_NO_ACTIVE_PROFILE == (uint
                )e.ErrorCode)
                {
                    System.Console.WriteLine("The printer
does not have an active profile");
                }
                else
                {
                    // more serious error,
                    throw e;
                }
            }

            //add a new profile
            pNova.AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC,
            out _newProfileID);
            //load the new profile
            pNova.LoadProfile(_newProfileID);
        }
    }
}
Hello document

// and set some options
pNova.SetOptionString(NOVAPDF_INFO_SUBJECT, "C#

// save profile
pNova.SaveProfile();
// set the copy profile as active profile ...
pNova.SetActiveProfile(_newProfileID);
// set nova default printer
pNova.SetDefaultPrinter();
// initialize OLE usage in novaPDF
pNova.InitializeOLEUsage("Word.Application");
// create Word application object
// license OLE server in novaPDF
pNova.LicenseOLEServer();
// initializations
object objMissing = System.Reflection.Missing
.Value;
object objTrue = true; object objFalse = false;
object strFile = @"C:\temp\test.docx";
// create Word document object
// print document
WordApp.ActivePrinter = PRINTER_NAME;
WordDoc.PrintOutOld(ref objFalse, ref objFalse, ref objMissing, ref objMissing, ref objMissing, ref objMissing, ref objMissing, ref objMissing, ref objMissing, ref objMissing, ref objMissing, ref objMissing, ref objMissing, ref objMissing, ref objMissing, ref objFalse);
// close Word objects
WordDoc.Close(ref objFalse, ref objMissing, ref objMissing, ref objFalse);
WordApp.Quit(ref objFalse, ref objMissing, ref objFalse);
WordApp = null;
}
catch (System.Runtime.InteropServices.COMException e)
{
    MessageBox.Show(e.Message);
}
catch (Exception e)
{
    MessageBox.Show(e.Message);
}

// restore active profile
if (!((activeProfile.Length > 0) && (activeProfile.CompareTo(PROFILE_NAME) != 0)))
{
    pNova.SetActiveProfile(activeProfile);
1.5.5.4 Convert Office Docs

The **Convert Office Docs** sample is a simple Windows console application that converts some Microsoft Office documents (Word, Excel, Publisher, PowerPoint and Visio) using Convert... functions. It uses Office OLE automation so Microsoft Office has to be installed.

The sample documents converted are located in the "OfficeDocuments" folder in novaPDF SDK installation folder. The pdf files are saved in the same folder.

Hidden links and internal document links are converted in pdf hyperlinks (except for Excel documents).

**Source code**

```csharp
using System;
using System.Windows.Forms;
// the novapiLib package must be added as a COM reference
using novapiLib11;
using Globals;

namespace Hello_World_CSharp
{
    class Class1
    {
        public static string PRINTER_NAME = "novaPDF SDK 11";
        public static string PROFILE_NAME = "ConvertOffice";
        public static int PROFILE_IS_PUBLIC = 0;

        [STAThread]
        static void Main(string[] args)
        {
            try
            {
                // create the NovaPdfOptions object
                NovaPdfOptions11 pNova = new NovaPdfOptions11();
                // initialize the NovaPdfOptions object
                // if you have an application license for novaPDF SDK,
                // pass the license key to the Initialize() function
                pNova.Initialize(PRINTER_NAME, "");

                // get the active profile ...
                string activeProfile = null;
                try
                {
                    pNova.GetActiveProfile(out activeProfile);
                }
                catch (System.Runtime.InteropServices.COMException e)
                {
                    // ignore profile exists error
                }
            }
        }
    }
}
```
if (NovaErrors.NV_NO_ACTIVE_PROFILE == (uint
)e.ErrorCode)
{
    System.Console.WriteLine("The printer does not have
an active profile");
}
else
{
    // more serious error, propagate it
    throw e;
}

string _newProfileID = String.Empty;
//add a new profile
pNova.AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC, out
_newProfileID);

//load the new profile
pNova.LoadProfile(_newProfileID);

//DO NOT OPEN PDF
//an Open action is added by default in the profile -
delete or disable this open action

//pNova.DisableActionType((int)ActionType.NOVA_ACTION_OPEN);

// set save options
nova.NovaTools.AddSaveOptions(pNova);

//save the new added profile
pNova.SaveProfile();

//set the new profile as the active profile
pNova.SetActiveProfile(_newProfileID);

//Path for novaPDF 11 SDK sample documents
string strDocumentsPath = "C:\Users\Public\Documents\novaPDF 11\SDK\OfficeDocuments\"
string strDocPath;

// -----------------------------------------------
// Convert Word document
// Microsoft Office Word has to be installed
// -----------------------------------------------
//Document file path
strDocPath = strDocumentsPath + "Word.docx";
//style name | level | type
string strHeadings = "Heading 1|1|0;Heading 2|2|0;Heading
3|3|0";

pNova.LicenseApplication("WINWORD.EXE");
pNova.LicenseApplication("SPLNOW64.EXE");
pNova.ConvertWordDocument(strDocPath, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 3, strHeadings);

// -----------------------------------------------
// Convert PowerPoint document
// Microsoft Office PowerPoint has to be installed
// -----------------------------------------------
//Document file path
1.5.5.5 Override Options

Override Options sample is a simple Windows console application that demonstrates the basic use of the INovaPDFOptions interface, without using profiles options. Instead of creating a profile and setting options in the profile, the sample uses SetOverrideOptions... functions to quickly set some...
options that will overwrite active profile settings. This is useful if you have a simple application where you just wish to change the pdf file name, email address or other simple options for each printed document.

There is a limited set of options that can be set with SetOverrideOptions... functions, usually settings that users can also change from the Save As dialog or the other prompt dialogs that can be shown before printing a document. If you need more complex settings, like adding a watermark, overlay or signature you have to configure them in a profile.

You can combine usage of profiles and override options functions. For instance you can create a profile with a watermark that is set as active and then use the override function just to change the file name for each document.

Once set, override options are taken in account for all printed documents until they are removed with DeleteOverrideOptions function.

The sample prints one page with a simple text to the novaPDF SDK 11 and generates a "OverrideOptions.pdf" file in the C:\temp folder.

Source code

```csharp
using System;
using System.Drawing;
using System.Drawing.Printing;
using System.Windows.Forms;
using novapiLib;
using Globals;

namespace Override_Options{
    ///<summary>
    ///Summary description for Class1.
    ///</summary>
    class Class1
    {
        /// <summary>
        /// The main entry point for the application.
        ///</summary>
        public static string PRINTER_NAME = "novaPDF SDK 11";
        public static string PDF_TEXT = "Override Options";
        public static string PDF_FILE_NAME = "OverrideOptions.pdf";

        [STAThread]
        static void Main(string[] args)
        {
            try
            {
                // create the NovaPdfOptions object
                NovaPdfOptions11 pNova = new NovaPdfOptions11();
                // initialize the NovaPdfOptions object
                // if you have an application license for novaPDF SDK,
                // pass the license key to the Initialize() function
                pNova.Initialize(PRINTER_NAME, "");
            } catch (Exception e) {
                // handle exceptions
            }
        }
    }
}
```
// Override options from active profile

// Save options
// do not show prompt save dialog
pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_SAVE_TYPE, (int)SaveDlgType.PROMPT_SAVE_NONE);
// save the pdf in next folder
pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_SAVE_FOLDER_TYPE, (int)SaveFolder.SAVEFOLDER_CUSTOM);
// set to insert at page number
pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_SAVE_FOLDER, "C:\\temp");
// pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_SAVE_FILE, PDF_FILE_NAME);
// access folder, if restricted
pNova.SetOverrideOptionEncryptedString(NovaOptions.NOVAPDF_OVERRIDE_SAVE_USER, "user");
// pNova.SetOverrideOptionEncryptedString(NovaOptions.NOVAPDF_OVERRIDE_SAVE_PASSWORD, "password");
// if a file with the same name already exists
pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_SAVE_PASSWORD, (int)SaveFileConflictType.FILE_CONFLICT_STRATEGY_OVERWRITE);

// disable open PDF in viewer

// pNova.SetOverrideOptionBool(NovaOptions.NOVAPDF_OVERRIDE_OPEN_VIEWER, 0);

// Embed fonts
pNova.SetOverrideOptionBool(NovaOptions.NOVAPDF_OVERRIDE_EMBEDFONTS, 1);

// Set compression level

// pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_COMPRESSION, (int)CompressionOverride.COMPRESSION_HIGHQUALITY);

// Merge PDF
// Enable merge
//
// pNova.SetOverrideOptionBool(NovaOptions.NOVAPDF_OVERRIDE_MERGE_PDF, 1);
// Merge with the same file, if the file exists in destination OR merge with another file
pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_MERGE_FILE_TYPE, (int)MergeFile.MERGE_FILE_OTHER);
// Append to existing PDF, insert at the beginning or insert at a page number
pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_MERGE_TYPE, (int)MergeType.MERGE_TYPE_APPEND);
// if merge with other file, specify file name (and path)

// pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_MERGE_FILE, "invoice-text.pdf");
// if merge type is set to insert at page number

// pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_INSERT_PAGENO, 1);
//if the existing PDF is encrypted

//pNova.SetOverrideOptionEncryptedString(NovaOptions.NOVAPDF_OVERRIDE_P
ASSWORD, "password");
/*

//Document info
pNova.SetOverrideOptionBool(NovaOptions .NOVAPDF_OVERRIDE_DOC_INFO, 1);
pNova.SetOverrideOptionString(NovaOptions .NOVAPDF_OVERRIDE_DOC_TITLE, "sdk title");
pNova.SetOverrideOptionString(NovaOptions .NOVAPDF_OVERRIDE_DOC_SUBJECT, "sdk subject");
pNova.SetOverrideOptionString(NovaOptions .NOVAPDF_OVERRIDE_DOC_AUTHOR, "sdk author");
pNova.SetOverrideOptionString(NovaOptions .NOVAPDF_OVERRIDE_DOC_KEY, "sdk key");
pNova.SetOverrideOptionString(NovaOptions .NOVAPDF_OVERRIDE_DOC_CREATOR, "sdk creator");

//Encrypt PDF
*/

pNova.SetOverrideOptionBool(NovaOptions.NOVAPDF_OVERRIDE_SECURITY, 1);

pNova.SetOverrideOptionEncryptedString(NovaOptions.NOVAPDF_OVERRIDE_SECURITY_Y_U, "user");

pNova.SetOverrideOptionEncryptedString(NovaOptions.NOVAPDF_OVERRIDE_SECURITY_Y_O, "owner");
/*

//Email

pNova.SetOverrideOptionBool(NovaOptions .NOVAPDF_OVERRIDE_SEND_EMAIL, 1);
pNova.SetOverrideOptionString(NovaOptions .NOVAPDF_OVERRIDE_REC_TO, "to");
pNova.SetOverrideOptionString(NovaOptions .NOVAPDF_OVERRIDE_REC_FROM, "from");
pNova.SetOverrideOptionString(NovaOptions .NOVAPDF_OVERRIDE_REC_CC, "cc");
pNova.SetOverrideOptionString(NovaOptions .NOVAPDF_OVERRIDE_REC_BCC, "bcc");

pNova.RegisterNovaEvent(NovaEvents.NOVAPDF_EVENT_ACTIONS);

// print a test page, using the previously set active profile settings
using (PrintDocument pd = new PrintDocument())
{
    pd.PrinterSettings.PrintName = PRINTER_NAME;
    pd.PrintPage += new PrintPageEventHandler
    (PrintPageFunction);
    pd.Print();
}

int bTimeout;
pNova.WaitForNovaEvent(120000, out bTimeout);
```csharp
        catch (System.Runtime.InteropServices.COMException e)
        {
            MessageBox.Show(e.Message);
        }
        catch (Exception e)
        {
            MessageBox.Show(e.Message);
        }
    }

    // find a predefined paper size
    private static PaperSize GetPageSize(string name)
    {
        PaperSize size = null;
        Name = Name.ToUpper();
        PrinterSettings settings = new PrinterSettings();
        foreach (PaperSize size in settings.PaperSizes)
        {
            if (size.Kind.ToString().ToUpper() == Name)
            {
                size = size;
                break;
            }
        }
        return size;
    }

    // and finally the function that actually prints the page
    private static void PrintPageFunction(object sender,
        PrintPageEventArgs ev)
    {
        string str = "novaPDF says Hello World from C#";
        Font font = new Font("Arial", 16);
        Brush brush = new SolidBrush(Color.Black);
        ev.Graphics.DrawString(str, font, brush, 20.0f, 20.0f);
        ev.HasMorePages = false;
    }
}
```

### 1.5.6 C++

#### 1.5.6.1 Hello World

**Hello World** sample is a simple Windows console application that prints one page with the "Hello World" text to the novaPDF SDK 11.

It demonstrates the basic use of the INovaPDFOptions interface. The printing job is made with Windows API calls OpenPrinter, StartDoc,...

It generates a "Hello World.pdf" file in the working folder.

**Notice**

If you do not use Windows API calls to print to novaPDF SDK 11, but you perform a print job by calling other controls "Print()" method, or if you print an existing document using "ShellExecute()" function, you should check the MFC Converter sample instead.

**Source code**
// HelloWorld.cpp
#include "stdafx.h"

//Include novaPDF headers
#include ".\..\include\novaOptions.h"
#include ".\..\include\novapi.h"

//name of novaPDF SDK 11
#define PRINTER_NAME L"novaPDF SDK 11"

//text to be written in the PDF file
#define PDF_TEXT L"Hello world!"

//PDF file name
#define PDF_FILE_NAME L"HelloWorld.pdf"

//Print profile name
#define PROFILE_NAME L"HelloWorld Profile"
#define PROFILE_IS_PUBLIC 0

//entry point for the console application
int _tmain(int argc, _TCHAR* argv[]) {
    HRESULT hr = S_OK;

    //initialize COM
    hr = CoInitialize(NULL);
    if (FAILED(hr))
    {
        MessageBox(NULL, L"Failed to initialize COM", L"novaPDF", MB_OK);
        return hr;
    }

    //create one NovaPdfOptions instance
    INovaPdfOptions *pNova = 0;
    hr = CoCreateInstance(__uuidof(NovaPdfOptions11), NULL, CLSCTX_INPROC_SERVER,
        if (FAILED(hr))
        {
            MessageBox(NULL, L"Failed to create novaPDF COM object", L"novaPDF", MB_OK);
            return hr;
        }

    //initialize the NovaPdfOptions object to use with a printer licensed for SDK
    hr = pNova->Initialize(PRINTER_NAME, L"");

    if (SUCCEEDED(hr))
    {
        LPWSTR pwsOldActiveProfileID = NULL;
        LPWSTR pwsNewProfileID = NULL;
        hr = pNova->GetActiveProfile(&pwsOldActiveProfileID);

        //create a new profile with default settings
        hr = pNova->AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC, &pwsNewProfileID);
//load the newly created profile
if (SUCCEEDED(hr) && pwsNewProfileID)
{
    hr = pNova->LoadProfile(pwsNewProfileID);
} else
{
    MessageBox(NULL, L"Failed to create profile", L"novaPDF", MB_OK);
    return hr;
}

// set PDF document Title
pNova->SetOptionString(NOVAPDF_DOCINFO_TITLE, L"My Doc Title");

//what type of save dialog to be shown (none, simple or ...
int dmFields = DM_PAPERSIZE | DM_PAPERLENGTH;
pDevmode->dmFields = dmFields;

//you may choose a predefined folder like "My Documents"
pNova->SetOptionLong(NOVAPDF_SAVE_FOLDER_TYPE, SAVEFOLDER_MYDOCUMENTS);

//set a macro for the file name
pNova->SetOptionString(NOVAPDF_SAVE_FILE_NAME, L"test_[N]");

//in case a file with the same name exists in the selected folder,
//choose what to do
pNova->SetOptionLong(NOVAPDF_SAVE_FILEEXIST_ACTION, FILE_CONFLICT_STRATEGY_AUTONUMBER_NEW);

//save profile changes
hr = pNova->SaveProfile();

//set as active profile for printer
pNova->SetActiveProfile(pwsNewProfileID);

HANDLE hPrinter;
PDEVMODE pDevmode = NULL;
PRINTER_DEFAULTS pd = { NULL, NULL, PRINTER_ACCESS_USE };

//start print job
if (OpenPrinter(PRINTER_NAME, &hPrinter, &pd))
{
    //get default printer DEVMODE
    int nSize = DocumentProperties(NULL, hPrinter, PRINTER_NAME, NULL, NULL, 0);
    pDevmode = (PDEVMODE)LocalAlloc(LPTR, nSize);
    DocumentProperties(NULL, hPrinter, PRINTER_NAME, pDevmode, NULL, DM_OUT_BUFFER);

    //set page size in DEVMODE
    pDevmode->dmPageSize = DMPAPER_USER;
    pDevmode->dmPaperLength = 2970; //5940;
    pDevmode->dmPaperWidth = 2100; //4200;
    pDevmode->dmFields = DM_PAPERSIZE | DM_PAPERLENGTH |
DM_PAPERWIDTH;

DocumentProperties(NULL, pInfo, PRINTER_NAME, pInfo, pInfo, DM_IN_BUFFER | DM_OUT_BUFFER);

// Print a page
HDC hDC = CreateDC(L"", PRINTER_NAME, NULL, pInfo);
DOCINFO docInfo = {sizeof(DOCINFO)};
// document name
docInfo.lpszDocName = PDF_FILE_NAME;
StartDoc(hDC, &docInfo);
StartPage(hDC);
// Draw text on page
TextOut(hDC, 100, 80, PDF_TEXT, (int)wcslen(PDF_TEXT));
EndPage(hDC);
EndDoc(hDC);
DeleteDC(hDC);

// Print job sent to printer
MessageBox(NULL, L"Print job finished", L"novaPDF", MB_OK);
LocalFree(pInfo);
ClosePrinter(hPrinter);
}
else
{
    WCHAR wsMessage[255];
    wsprintf(wsMessage, L"OpenPrinter failed, error = %d", GetLastError());
    MessageBox(NULL, wsMessage, L"novaPDF", MB_OK);
}

// restore default profile
if (pwsOldActiveProfileID)
{
    pNova->SetActiveProfile(pwsOldActiveProfileID);
}

// delete newly created profile
pNova->DeleteProfile(pwsNewProfileID);
// free memory
CoTaskMemFree(pwsNewProfileID);
CoTaskMemFree(pwsOldActiveProfileID);
}
else
{
    MessageBox(NULL, L"Failed to initialize novaPDF Printer", L"novaPDF", MB_OK);
}

// release NovaPdfOptions
pNova->Release();
CoUninitialize();
return 0;
1.5.6.2 Hello World (network)

Hello World (network) sample is similar with Hello World sample but it can be used from network and print to novaPDF installed on a different computer. The sample requests the shared printer name in a dialog as "\computer name\printer name" (for example if novaPDF SDK 11 is installed on WS1, then you should enter "\WS1\novaPDF SDK 11"). See Network use topic for more details on how to use novaPDF on network.

Source Code snippets (in addition to Hello World source code)

```c
{
    //initialize COM
    hr = CoInitialize(NULL);
    if (FAILED (hr))
    {
        MessageBoxW(NULL, L"Failed to initialize COM", L"novaPDF", MB_OK);
        return hr;
    }

    //create one NovaPdfOptions instance
    INovaPdfOptions *pNova = 0;
    hr = CoCreateInstance(__uuidof(NovaPdfOptions11), NULL, CLSCTX_INPROC_SERVER,
        "use create novaPDF COM object", L"novaPDF", MB_OK);
    return hr;
}

    DWORD dwSize = 256;
    WCHAR strWorkStation[256];

    //find out computer name
    GetComputerNameW(strWorkStation, &dwSize);
    PrinterNameDlg dlg;

    //construct printer name as "\computer name\printer name"
   dlg.m_strPrinterName.Format(L"\\%s\%s", strWorkStation, PRINTER_NAME);

    //get printer name from user
    if (IDCANCEL == dlg.DoModal())
    {
        pNova->Release();
        CoUninitialize();
        return hr;
    }
    CString strPrinterName = dlg.m_strPrinterName;

    //initialize the NovaPdfOptions object to use with a printer licensed for SDK
    hr = pNova->Initialize((LPCWSTR)strPrinterName, L"");

    //...
}
```
1.5.6.3 Convert Office Docs

The **Convert Office Docs** sample is a simple Windows console application that converts some Microsoft Office documents (Word, Excel, Publisher, PowerPoint and Visio) using Convert... functions. It uses Office OLE automation so Microsoft Office has to be installed.

The sample documents converted are located in the "OfficeDocuments" folder in novaPDF SDK installation folder. The pdf files are saved in the same folder.

Hidden links and internal document links are converted in pdf hyperlinks (except for Excel documents).

**Source code**

```c
#include "stdafx.h"
#include ..\\..\\..\include\novaOptions.h"
#include ..\\..\\..\include\novaEvents.h"
#include ..\\..\\..\include\novapi.h"
#include "nova.h"

#define PRINTER_NAME L"novaPDF SDK 11"

#define PROFILE_NAME L"ConvertOfficeDocs Profile"
#define PROFILE_IS_PUBLIC 0

int _tmain(int argc, _TCHAR* argv[]) {
    HRESULT hr = S_OK;

    //initialize COM
    hr = CoInitialize(NULL);
    if (FAILED (hr)) {
        MessageBox(NULL, L"Failed to initialize COM", L"novaPDF", MB_OK);
        return hr;
    }

    //create one NovaPdfOptions instance
    INovaPdfOptions11 *pNova = 0;
    hr = CoCreateInstance(__uuidof(NovaPdfOptions11), NULL,
        CLSCTX_INPROC_SERVER, __uuidof(INovaPdfOptions11), (LPVOID*) &pNova);
    if (FAILED(hr)) {
        MessageBox(NULL, L"Failed to create novaPDF COM object", L"novaPDF",
            MB_OK);
        return hr;
    }

    // initialize the NovaPdfOptions object to use with a printer licensed for
```
SDK

// if you have an application license for novaPDF SDK,
// pass the license key to the Initialize() function
hr = pNova->Initialize(PRINTER_NAME, L"\n");

if (SUCCEEDED(hr))
{
    LPWSTR pwsOldActiveProfileID = NULL;
    LPWSTR pwsNewProfileID = NULL;

    hr = pNova->GetActiveProfile(&pwsOldActiveProfileID);

    // create a new profile with default settings
    hr = pNova->AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC,
                        &pwsNewProfileID);

    // load the newly created profile
    if (SUCCEEDED(hr) && pwsNewProfileID)
    {
        hr = pNova->LoadProfile(pwsNewProfileID);
    }
    else
    {
        MessageBox(NULL, L"Failed to create profile", L"novaPDF",
                        MB_OK);
        return hr;
    }

    // set novaPDF options
    // uncomment the function calls for the options you wish to set and
    change the options in nova.cpp unit

    //DO NOT OPEN PDF
    //an Open action is added by default in the profile - delete or
disable this open action
    pNova->DisableActionType(NOVA_ACTION_OPEN);

    //Save pdf files to OfficeDocuments folder
    AddSaveOptions(pNova);

    // save profile changes
    hr = pNova->SaveProfile();

    // set as active profile for printer
    pNova->SetActiveProfile(pwsNewProfileID);

    //Path for novaPDF 11 SDK sample documents
    WCHAR wsDocumentsPath[MAX_PATH];
    WCHAR wsDocPath[MAX_PATH];
    SHGetFolderPath(NULL, CSIDL_COMMON_DOCUMENTS, NULL, 0,
                              wsDocumentsPath);
    wcscat(wsDocumentsPath, L"\novaPDF 11\SDK\OfficeDocuments\");

    // Convert Word document
// Microsoft Office Word has to be installed
//@begin example 1
wstring wsDocPath, wsDocumentsPath;
wcscpy(wsDocPath, L"Word.docx");
wcscat(wsDocPath, L"Word.docx");
//@end example 1

// Document file path
wscopy(wsDocPath, wsDocumentsPath);
wscat(wsDocPath, L"WINWORD.EXE");
pNova->LicenseApplication(L"WINWORD.EXE");
hr = pNova->ConvertWordDocument(wsDocPath, TRUE, FALSE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, wsHeadings);

// Convert PowerPoint document
//@begin example 2
wscopy(wsDocPath, wsDocumentsPath);
wscat(wsDocPath, L"PowerPoint.pptx");
pNova->LicenseApplication(L"POWERPNT.EXE");
pNova->LicenseApplication(L"SPLWOW64.EXE");
hr = pNova->ConvertPowerPointDocument(wsDocPath, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE);
//@end example 2

// Convert Publisher document
//@begin example 3
wscopy(wsDocPath, wsDocumentsPath);
wscat(wsDocPath, L"Publisher.pub");
pNova->LicenseApplication(L"MSPUB.EXE");
pNova->LicenseApplication(L"SPLWOW64.EXE");
hr = pNova->ConvertPublisherDocument(wsDocPath, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE);
//@end example 3

// Convert Excel document
//@begin example 4
wscopy(wsDocPath, wsDocumentsPath);
wscat(wsDocPath, L"Excel.xlsx");
pNova->LicenseApplication(L"EXCEL.EXE");
pNova->LicenseApplication(L"SPLWOW64.EXE");
hr = pNova->ConvertExcelDocument(wsDocPath, TRUE);
//@end example 4

// Convert Visio document
//@begin example 5
wscopy(wsDocPath, wsDocumentsPath);
//@end example 5
wcscat(wsDocPath, L"Visio.vsd");
pNova->LicenseApplication(L"VISIO.EXE");
pNova->LicenseApplication(L"SPLWOW64.EXE");
hr = pNova->ConvertVisioDocument(wsDocPath, TRUE, TRUE, TRUE, TRUE, TRUE);

//restore default profile
pNova->SetActiveProfile(pws0ldActiveProfileID);

//delete newly created profile
pNova->DeleteProfile(pwsNewProfileID);
//free memory
CoTaskMemFree(pwsNewProfileID);
if(pws0ldActiveProfileID)
{
    CoTaskMemFree(pws0ldActiveProfileID);
}
else
{
    MessageBox(NULL, L"Failed to initialize novaPDF Printer", L"novaPDF", MB_OK);
}

//release NovaPdfOptions
pNova->Release();
CoUninitialize();
return 0;

1.5.6.4 MFC Converter

The **MFC Converter** sample demonstrates how to convert an existing file by printing it to novaPDF SDK 11 using the ShellExecute function. It also demonstrates how to set different options and manage profiles.

The same approach should be taken if you print using a "Print()" method from another object (like an internet browser or a report control). Just replace the ShellExecute call with the call of your Print method.

When the application starts, it creates a few profiles and makes different settings in the profiles. Then it shows a dialog from where the user can select the active profile and change its settings using the controls from the dialog.

After that a document can be selected from the harddisk and printed to novaPDF SDK 11 using the ShellExecute function call.

When using this technique to convert a file to PDF, you have to take care of the fact that ShellExecute prints to the default printer. This function returns immediately and does not wait until the print is
finished (it may return before the printing is actually started). Therefore you have to set the default printer to novaPDF SDK 11 before calling ShellExecute (using the SetDefaultPrinter method), register FileSaved message (or any other novaPDF SDK 11 message) to be sure that the print job was started. In this message handler restore the default printer (with the RestoreDefaultPrinter method). This way you made sure that the default printer was restored and your document is printed to novaPDF SDK 11.

Source code snippets

1. Declare INovaPdfOptions variable

   //declare an INovaPdfOptions member variable
   private:
     INovaPdfOptions *m_novaOptions;

2. Register novaPDF SDK 11 messages

   const UINT  wm_Nova_FileSaved = RegisterWindowMessageW( MSG_NOVAPDF2_FILESAVED );
   const UINT  wm_Nova_PrintError = RegisterWindowMessageW( MSG_NOVAPDF2_PRINTERROR )

   BEGIN_MESSAGE_MAP(CnovaPrintDlg, CDialog)
     //...
     ON_REGISTERED_MESSAGE(wm_Nova_FileSaved, OnNovaPDFFileSaved)
     ON_REGISTERED_MESSAGE(wm_Nova_PrintError, OnNovaPDFPrintError)
     //...
   END_MESSAGE_MAP()

3. Initialize INovaPdfOptions

   BOOL CnovaPrintDlg::OnInitDialog()
   {
     //...
     HRESULT hr = S_OK;
     m_novaOptions = 0;

     //create an instance of INovaPdfOptions
     hr = CoCreateInstance(__uuidof(NovaPdfOptions11), NULL, CLSCTX_INPROC_SERVER, __uuidof(INovaPdfOptions11), (LPVOID*) L"");
     if (SUCCEEDED(hr)) {
       // initialize the NinjaPdfOptions object to use with a printer licensed for NinjaPDF
       hr = m_novaOptions->Initialize(PRINTER_NAME, L"");
     }
     else {
       ::MessageBoxW(NULL, L"Failed to create novaPDF COM object", L"novaPDF", MB_OK);
     }
     //...
   }

4. Release INovaPDFOptions

   CnovaPrintDlg::~CnovaPrintDlg()
   {
     //...
   }
/delete profiles
if (m_wsProfileSmall)
{
    hr = m_novaOptions->DeleteProfile(m_wsProfileSmall);
    CoTaskMemFree(m_wsProfileSmall);
}
if (m_wsProfileFull)
{
    hr = m_novaOptions->DeleteProfile(m_wsProfileFull);
    CoTaskMemFree(m_wsProfileFull);
}
// destroy our nova options object
if (m_novaOptions) {
    m_novaOptions->Release();
}
// uninitialize COM libraries
CoUninitialize();
//...

5. Set novaPDF SDK 11 Options
BOOL CnovaPrintDlg::OnInitDialog()
{
    // Add a profile called "Small size". If profile L"Small size" exists this will fail
    hr = m_novaOptions->AddProfile(SMALL_SIZE_PROFILE, PROFILE_IS_PUBLIC, &m_wsProfileSmall);

    //load the newly created profile
    if (SUCCEEDED(hr) && m_wsProfileSmall)
    {
        //load profile
        m_novaOptions->LoadProfile(m_wsProfileSmall);

        // disable the "Save PDF file as" prompt
        m_novaOptions->SetOptionLong(NOVAPDF_SAVE_PROMPT_TYPE, PROMPT_SAVE_NONE);
        // set generated Pdf files destination folder ("c:\")
        m_novaOptions->SetOptionLong(NOVAPDF_SAVE_LOCATION, LOCATION_TYPELOCAL);
        m_novaOptions->SetOptionLong(NOVAPDF_SAVE_FOLDER_TYPE, SAVEFOLDER_CUSTOM);
        m_novaOptions->SetOptionString(NOVAPDF_SAVE_FOLDER, szExeDirectory);
        // set output file name
        m_novaOptions->SetOptionString(NOVAPDF_SAVE_FILE_NAME, L"PDF Converter small size.pdf");
        // if file exists in the destination folder, append a counter to the end of the file name
        m_novaOptions->SetOptionLong(NOVAPDF_SAVE_FILEEXIST_ACTION, FILE_CONFLICT_STRATEGY_AUTONUMBER_NEW);
        //Set other options
        //...
        //save profile changes
        m_novaOptions->SaveProfile();
6. Start a print job
   
   ```
   void CnovaPrintDlg::OnBnClickedOk()
   {
     //...
     HRESULT hr = S_OK;

     // set the active profile to be used for printing
     hr = m_novaOptions->SetActiveProfile(m_strProfileId);

     // register our window to receive messages from the printer
     hr = m_novaOptions->RegisterEventWindow((LONG) GetSafeHwnd());

     // set novaPDF as default printer, so it will be used by ShellExecute
     hr = m_novaOptions->SetDefaultPrinter();

     // license file for ShellExecute
     hr = m_novaOptions->LicenseShellExecuteFile(m_strFileToConvert.AllocSysString());

     // print the document
     m_bPrintJobPending = TRUE;
     HINSTANCE hExec = ShellExecute(GetSafeHwnd(), L"print", m_strFileToConvert, NULL, NULL, SW_HIDE);

     //...
   }
   ```

7. Restore default printer when printing finished

   ```
   LRESULT CnovaPrintDlg::OnNovaPDFFileSaved(WPARAM wParam, LPARAM lParam)
   {
     // restore original default printer
     m_novaOptions->UnRegisterEventWindow();
     m_novaOptions->RestoreDefaultPrinter();
     m_bPrintJobPending = FALSE;
     return 0;
   }
   ```

   ```
   LRESULT CnovaPrintDlg::OnNovaPDFPrintError(WPARAM wParam, LPARAM lParam)
   {
     switch(wParam){
     case ERROR_MSG_TEMP_FILE:
       MessageBox(L"Error saving temporary file on printer server", L"novaPDF", MB_OK);
       break;
     case ERROR_MSG_LIC_INFO:
       MessageBox(L"Error reading license information", L"novaPDF", MB_OK);
       break;
     case ERROR_MSG_SAVE_PDF:
       MessageBox(L"Error saving PDF file", L"novaPDF", MB_OK);
       break;
     case ERROR_MSG_JOB_CANCELED:
       MessageBox(L"Print job was canceled", L"novaPDF", MB_OK);
       break;
     }
1.5.6.5 MFC Scribble

The MFC Scribble sample extends the standard MFC Scribble sample with the generation of PDF files using novaPDF SDK 11. It demonstrates how to integrate novaPDF SDK in a document/view MFC architecture:

**Source code snippets**

1. Register novaPDF SDK 11 event

```c
#define PROFILE_NAME L"MFCScribble Profile"
#define PDF_FILE_NAME L"MFCScribble.pdf"
#define PROFILE_IS_PUBLIC 0

// This message is sent when the PDF file is finished and saved on the harddisk
const UINT wm_Nova_FileSaved = RegisterWindowMessageW(MSG_NOVAPDF2_FILESAVED);
BEGIN_MESSAGE_MAP(CScribbleView, CScrollView)
    ON_REGISTERED_MESSAGE(wm_Nova_FileSaved, OnNovaPDFFileSaved)
END_MESSAGE_MAP()
```

2. Initialize INovaPDFOptions

```c
CScribbleView::CScribbleView()
{
    //...    HRESULT hr = CoInitialize(NULL);
    //...    //create novaPDFOptions object
    hr = CoCreateInstance(__uuidof(NovaPdfOptions11), NULL, CLSCTX_INPROC_SERVER, __uuidof(INovaPDFOptions11), (LPVOID*)...the NovaPdfOptions object to use with a printer licensed for SDK
    hr = m_pNova->Initialize(PRINTER_NAME, L"");
    //...    
}
```

3. Release INovaPDFOptions

```c
CScribbleView::~CScribbleView()
{
    //release novaPDFOptions object
    if (m_pNova){
        m_pNova->Release();
    }
    CoUninitialize();
}
```

4. Set novaPDF SDK 11 Options

```c
BOOL CScribbleView::OnPreparePrinting(CPrintInfo* pInfo)
```
{
    //set novaPDF default printer
    if (m_pNova) {
        m_pNova->SetDefaultPrinter();
        HRESULT hr;
        hr = m_pNova->GetActiveProfile(&m_wsDefaultProfile);

        //create a new profile with default settings
        hr = m_pNova->AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC, &m_wsNewProfile);

        //load the newly created profile
        if (SUCCEEDED(hr) && m_wsNewProfile) {
            hr = m_pNova->LoadProfile(m_wsNewProfile);
        } else {
            MessageBox(L"Failed to create profile", L"novaPDF", MB_OK);
            return hr;
        }

        // set PDF document Title
        m_pNova->SetOptionString(NOVPDF_DOCINFO_TITLE, L"MFC Scribble Sample");

        // set resulting file name
        m_pNova->SetOptionLong(NOVPDF_SAVE_FOLDER_TYPE, SAVEFOLDER_CUSTOM);
        m_pNova->SetOptionLong(NOVPDF_SAVE_LOCATION, LOCATION_TYPE_LOCAL);
        m_pNova->SetOptionLong(NOVPDF_SAVE_FOLDER_TYPE, SAVEFOLDER_CUSTOM);
        m_pNova->SetOptionString(NOVPDF_SAVE_FOLDER, L"C:\\temp");
        m_pNova->SetOptionString(NOVPDF_SAVE_FILE_NAME, PDF_FILE_NAME);

        //do not show prompt dialog
        m_pNova->SetOptionLong(NOVPDF_SAVE_PROMPT_TYPE, PROMPT_SAVE_NONE);

        //if file exists, override
        m_pNova->SetOptionLong(NOVPDF_SAVE_FILEEXIST_ACTION, FILE_CONFLICT_STRATEGY_OVERWRITE);

        //save profile changes
        hr = m_pNova->SaveProfile();

        //set as active profile for printer
        m_pNova->SetActiveProfile(m_wsNewProfile);

        //register window to receive messages from novaPDF printer
        m_pNova->RegisterEventWindow((LONG)m_hWnd);
        //...
    }
}

5. Restore options when printing finished
void CScribbleView::OnEndPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/) {
    if (m_pNova) {
        //unregister events
}
m_pNova->UnRegisterEventWindow();
//restore default profile
if (m_wsDefaultProfile)
{
    m_pNova->SetActiveProfile(m_wsDefaultProfile);
}

//delete newly created profile
m_pNova->DeleteProfile(m_wsNewProfile);
//free memory
CoTaskMemFree(m_wsNewProfile);
CoTaskMemFree(m_wsDefaultProfile);
m_wsDefaultProfile = NULL;
m_wsNewProfile = NULL;
//restore default printer
m_pNova->RestoreDefaultPrinter();
}

6. novaPDF SDK 11 message handler
LRESULT CScribbleView::OnNovaPDFFileSaved(WPARAM, LPARAM){
    //PDF is saved, so just show a message that the conversion to PDF was successful
    MessageBox(L"PDF file was saved successfully", L"novaPrint");
    return 0;
}

1.5.6.6 Temporary printer

Temporary printer sample is similar with Hello World sample but it uses temporary printers. It demonstrates the use of AddNovaPrinter and DeleteNovaPrinter.

Source code

#include "stdafx.h"

//Include novaPDF headers
#include "..\..\..\include\novaOptions.h"

//NovaPdfOptions
#include "..\..\..\include\novapi.h"

#include "nova.h"

//name of novaPDF Printer Demo
#define PRINTER_NAME    L"novaPDF temporary printer"
#define PORT_NAME    L"novaPDF11temp"

//text to be written in the PDF file
#define PDF_TEXT        L"Hello world!"

//PDF file name
#define PDF_FILE_NAME   L"HelloWorld.pdf"

//Print profile name
#define PROFILE_NAME     L"HelloWorld Profile"
#define PROFILE_IS_PUBLIC 0

//entry point for the console application
int _tmain(int argc, _TCHAR* argv[])
{
    HRESULT hr = S_OK;

    //initialize COM
    hr = CoInitialize(NULL);
    if (FAILED (hr))
    {
        MessageBox(NULL, L"Failed to initialize COM", L"novaPDF", MB_OK);
        return hr;
    }

    //create one NovaPdfOptions instance
    INovaPdfOptions *pNova = NULL;
    hr = CoCreateInstance(__uuidof(NovaPdfOptions11), NULL, CLSCTX_INPROC_SERVER, __uuidof(INovaPdfOptions11), (LPVOID*) &pNova);
    if (SUCCEEDED(hr))
    {
        //add temporary printer
        pNova->AddNovaPrinter(PRINTER_NAME, PORT_NAME, L"nPdfSdk11_Softland", L"8501", L"");
        // set optional PDF settings
        LPWSTR pwsNewProfileID = NULL;
        //create a new profile with default settings
        hr = pNova->AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC, &pwsNewProfileID);
    
        //load the newly created profile
        if (SUCCEEDED(hr) && pwsNewProfileID)
        {
            hr = pNova->LoadProfile(pwsNewProfileID);
        }
        else
        {
            MessageBox(NULL, L"Failed to create profile", L"novaPDF", MB_OK);
            return hr;
        }

        // set novaPDF options
        // set PDF document Title
        pNova->SetOptionString(NOVAPDF_DOCINFO_TITLE, L"My Doc Title");

        //what type of save dialog to be shown (none, simple or extended)
        pNova->SetOptionLong(NOVAPDF_SAVE_PROMPT_TYPE, PROMPT_SAVE_SIMPLE);

        //you may choose a predefined folder like "My Documents"
        pNova->SetOptionLong(NOVAPDF_SAVE_FOLDER_TYPE, SAVEFOLDER_MYDOCUMENTS);

        //set a macro for the file name
        pNova->SetOptionString(NOVAPDF_SAVE_FILE_NAME, L"test_[N]");
//in case a file with the same name exists in the selected folder, choose what to do
pNova->SetOptionLong(NOVAPDF_SAVE_FILEEXIST_ACTION,
FILE_CONFLICT_STRATEGY_AUTONUMBER_NEW);

//save profile changes
hr = pNova->SaveProfile();

//set as active profile for printer
pNova->SetActiveProfile(pwsNewProfileID);

HANDLE hPrinter;
PDEVMODE pDevmode = NULL;
PRINTER_DEFAULTS pd = { NULL, NULL, PRINTER_ACCESS_USE };

//start print job
if (OpenPrinter(PRINTER_NAME, &hPrinter, &pd))
{
    //get default printer DEVMODE
    int nSize = DocumentProperties(NULL, hPrinter,
PRINTER_NAME, NULL, NULL, 0);
    pDevmode = (PDEVMODE)LocalAlloc(LPTR, nSize);
    DocumentProperties(NULL, hPrinter, PRINTER_NAME,
    pDevmode, NULL, DM_OUT_BUFFER);

    //set page size in DEVMODE
    pDevmode->dmPageSize = DMPAPER_USER;
    pDevmode->dmPaperLength = 2970;
    pDevmode->dmPaperWidth = 2100;
    pDevmode->dmFields = DM_PAPERSIZE | DM_PAPERSIZE |
    DM_PAPERWIDTH;
    DocumentProperties(NULL, hPrinter, PRINTER_NAME,
    pDevmode, pDevmode, DM_IN_BUFFER | DM_OUT_BUFFER);

    //Print a page
    HDC hDC = CreateDC(L"", PRINTER_NAME, NULL, pDevmode);
    DOCINFO docInfo = {sizeof(DOCINFO)};
    // PDF document name and path
docInfo.lpszDocName = PDF_FILE_NAME;
    StartDoc(hDC, &docInfo);
    StartPage(hDC);
    // Draw text on page
    TextOut(hDC, 100, 80, PDF_TEXT, (int)wcslen(PDF_TEXT));
    EndPage(hDC);
    EndDoc(hDC);
    DeleteDC(hDC);

    //print job sent to printer
    MessageBox(NULL, L"Print job finished", L"novaPDF",
MB_OK);

    LocalFree(pDevmode);
    ClosePrinter(hPrinter);
}
else
{
    WCHAR wsMessage[255];
    wsprintf(wsMessage, L"OpenPrinter failed, error = %d",

GetLastError();
    MessageBox(NULL, wsMessage, L"novaPDF", MB_OK);
}

//delete newly created profile
pNova->DeleteProfile(pwsNewProfileID);
//free memory
CoTaskMemFree(pwsNewProfileID);
//delete temporarily printer
pNova->DeleteNovaPrinter(PRINTER_NAME);
}

//release NovaPdfOptions
pNova->Release();
CoUninitialize();
return 0;
}

1.5.6.7 Multiple printers

Multiple printers sample is similar with the Temporary printer sample but it uses several threads.

Source code

// HelloWorld.cpp
#include "stdafx.h"
//Include novaPDF headers
#include "..\..\..\include\novaOptions.h"

#include "..\..\..\include\novapi.h"
#include "nova.h"

//name of novaPDF Printer Demo
#define PRINTER_NAME1 L"novaPDF temporary printer1"
#define PRINTER_NAME2 L"novaPDF temporary printer2"
#define PRINTER_NAME3 L"novaPDF temporary printer3"
#define PORT_NAME1 L"novaPDF11temp1"
#define PORT_NAME2 L"novaPDF11temp2"
#define PORT_NAME3 L"novaPDF11temp3"
#define FILE_NAME1 L"first.pdf"
#define FILE_NAME2 L"second.pdf"
#define FILE_NAME3 L"third.pdf"

//text to be written in the PDF file
#define PDF_TEXT L"Hello world!"

//PDF file name
#define PDF_FILE_NAME L"HelloWorld.pdf"
//Print profile name
#define PROFILE_NAME L"HelloWorld Profile"
#define PROFILE_IS_PUBLIC 0

typedef struct _PRT_THREAD_PARAM {
    WCHAR wsPrinterName[255];
    WCHAR wsPortName[255];
    WCHAR wsFileName[255];
} PRT_THREAD_PARAM;

DWORD WINAPI PrtThreadProc(LPVOID lpParameter);
HANDLE CreatePrtThread(LPWSTR p_strPrinterName, LPWSTR p_wsPortName, LPWSTR p_wsFileName);

//entry point for the console application
int _tmain(int argc, _TCHAR* argv[])
{
    HANDLE hThread1 = CreatePrtThread(PRINTER_NAME1, PORT_NAME1, FILE_NAME1);
    HANDLE hThread2 = CreatePrtThread(PRINTER_NAME2, PORT_NAME2, FILE_NAME2);
    HANDLE hThread3 = CreatePrtThread(PRINTER_NAME3, PORT_NAME3, FILE_NAME3);

    if (hThread1 > 0){
        //wait to stop processing events
        WaitForSingleObject(hThread1, INFINITE);
        CloseHandle(hThread1);
    }

    if (hThread2 > 0){
        //wait to stop processing events
        WaitForSingleObject(hThread2, INFINITE);
        CloseHandle(hThread2);
    }

    if (hThread3 > 0){
        //wait to stop processing events
        WaitForSingleObject(hThread3, INFINITE);
        CloseHandle(hThread3);
    }

    return 0;
}

HANDLE CreatePrtThread(LPWSTR p_strPrinterName, LPWSTR p_wsPortName, LPWSTR p_wsFileName)
{
    PRT_THREAD_PARAM* pParams;
    DWORD dwThreadId;

    // Transmit parameters for thread: pipe handle and PDF temp file name
    pParams = (PRT_THREAD_PARAM*)GlobalAlloc(LPTR, sizeof(PRT_THREAD_PARAM));
    wcscpy(pParams->wsPrinterName, p_strPrinterName);
    wcscpy(pParams->wsPortName, p_wsPortName);
    wcscpy(pParams->wsFileName, p_wsFileName);

    // Create the thread
    return 0;
}
HANDLE hThread = CreateThread(  
    NULL, // no security attribute
    0,    // default stack size
    (LPTHREAD_START_ROUTINE) PrtThreadProc,  
    (LPVOID) pParams, // thread parameter
    0, // not suspended
    &dwThreadId); // returns thread ID
return hThread;
}

DWORD WINAPI PrtThreadProc(LPVOID lpParameter)
{
    HRESULT hr = S_OK;

    //initialize COM
    hr = CoInitialize(NULL);
    if (FAILED (hr))
    {
        MessageBox(NULL, L"Failed to initialize COM", L"novaPDF", MB_OK);
        return hr;
    }

    // Read thread's parameter: a handle to a pipe instance and the name of the temporary PDF file
    PRT_THREAD_PARAM* pParams = ((PRT_THREAD_PARAM*)lpParameter);

    INovaPdfOptions *pNova = NULL;
    hr = CoCreateInstance(__uuidof(NovaPdfOptions11), NULL,  
                          CLSCTX_INPROC_SERVER, __uuidof(INovaPdfOptions11), (LPVOID*) &pNova);

    //if you have an application license for novaPDF SDK, call the RegisterLicenseKey() function
    //hr = pNova->RegisterLicenseKey"<registration name>", "<license key>",  
    //"<application name>");

    if (SUCCEEDED(hr))
    {
        //add temporary printer
        pNova->AddNovaPrinter(pParams->wsPrinterName,  
            pParams->wsPortName, L"nPdfSdk11_Softland", L"8501", L""");

        // set optional PDF settings
        LPWSTR pwsNewProfileID = NULL;
        //create a new profile with default settings
        hr = pNova->AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC,  
            &pwsNewProfileID);

        //load the newly created profile
        if (SUCCEEDED(hr)) && pwsNewProfileID)
        {
            hr = pNova->LoadProfile(pwsNewProfileID);
        } else
        {
            // code to handle error
pNova->Release();
return hr;
}

if (SUCCEEDED(hr) && pwsNewProfileID)
{
    // set novaPDF options
    // set resulting file name
    pNova->SetOptionLong(NOAPDF_SAVE_FOLDER_TYPE, SAVEFOLDER_CUSTOM);
pNova->SetOptionLong(NOAPDF_SAVE_LOCATION, LOCATION_TYPE_LOCAL);
pNova->SetOptionLong(NOAPDF_SAVE_FOLDER_TYPE, SAVEFOLDER_CUSTOM);
pNova->SetOptionString(NOAPDF_SAVE_FOLDER, L "C:\\temp\\novaPDF");pNova->SetOptionString(NOAPDF_SAVE_FILE_NAME, pParams->wsFileName);
    // do not show prompt dialog
    pNova->SetOptionLong(NOAPDF_SAVE_PROMPT_TYPE, PROMPT_SAVE_NONE);
    // if file exists, override
    pNova->SetOptionLong(NOAPDF_SAVE_FILEEXIST_ACTION, FILE_CONFLICT_STRATEGY_AUTONUMBER_NEW);
    // do not open
    pNova->SetOptionBool(NOAPDF_ACTION_DEFAULT_VIEWER, FALSE);

    // save profile changes
    hr = pNova->SaveProfile();
    // set as active profile for printer
    pNova->SetActiveProfile(pwsNewProfileID);

    HANDLE hPrinter;
    BOOL bTimeout;
    PDEVMODE pDevmode = NULL;
    PRINTER_DEFAULTS pd = { NULL, NULL, PRINTER_ACCESS_USE };
    for (int i = 1; i <= 10; i++)
    {
        // start print job
        if (OpenPrinter(pParams->wsPrinterName, &hPrinter, &pd))
        {
            // register to wait for a nova event - wait
            pNova->RegisterNovaEvent(L "NOAPDF_EVENT_START_DOC");
            // get default printer DEVMODE
            int nSize = DocumentProperties(NULL, hPrinter, pParams->wsPrinterName, NULL, NULL, 0);
            pDevmode = (PDEVMODE)LocalAlloc(LPTR, nSize);
            DocumentProperties(NULL, hPrinter, pParams->wsPrinterName, pDevmode, NULL, DM_OUT_BUFFER);

            // set page size in DEVMODE
            pDevmode->dmPageSize = DMPAPER_USER;
            pDevmode->dmPaperLength = 2970; // 5940;
pDevmode->dmPaperWidth = 2100;//4200;
pDevmode->dmFields = DM_PAPERSIZE | DM_PAPERPITCH;

DM_PAPERLENGTH | DM_PAPERWIDTH;
DocumentProperties(NULL, hPrinter,
pParams->wsPrinterName, pDevmode, pDevmode, DM_IN_BUFFER | DM_OUT_BUFFER);

   //Print a page
   HDC hDC = CreateDC(L"",
pParams->wsPrinterName, NULL, pDevmode);
   DOCINFO docInfo = {sizeof(DOCINFO)};
   // PDF document name and path
docInfo.lpszDocName = PDF_FILE_NAME;
StartDoc(hDC,&docInfo);
StartPage(hDC);
   // Draw text on page
TextOut(hDC, 100, 80, PDF_TEXT, (int)
wcslen(PDF_TEXT));

   EndPage(hDC);
   EndDoc(hDC);
   DeleteDC(hDC);
   LocalFree(pDevmode);
   ClosePrinter(hPrinter);
   pNova->WaitForNovaEvent(-1, &bTimeout);

   }

   //delete profile
   pNova->DeleteProfile(pwsNewProfileID);
   CoTaskMemFree(pwsNewProfileID);

   //delete temporary printer
   pNova->DeleteNovaPrinter(pParams->wsPrinterName);

   }

   //release NovaPdfOptions
   pNova->Release();
   return 0;

1.5.6.8 Override Options

Override Options sample is a simple Windows console application that demonstrates the basic use of the INovaPDFOptions interface, without using profiles options. Instead of creating a profile and setting options in the profile, the sample uses SetOverrideOptions... functions to quickly set some options that will overwrite active profile settings. This is useful if you have a simple application where you just wish to change the pdf file name, email address or other simple options for each printed document.

There is a limited set of options that can be set with SetOverrideOptions... functions, usually settings that users can also change from the Save As dialog or the other prompt dialogs that can be shown before printing a document. If you need more complex settings, like adding a watermark, overlay or signature you have to configure them in a profile.

You can combine usage of profiles and override options functions. For instance you can create a profile with a watermark that is set as active and then use the override function just to change the
file name for each document.

Once set, override options are taken in account for all printed documents until they are removed with DeleteOverrideOptions function.

The sample prints one page with a simple text to the novaPDF SDK 11 and generates a "OverrideOptions.pdf" file in the C:\temp folder.

**Source code**

```cpp
#include "stdafx.h"
#include "../..\..\..\include\novaOptions.h"
#include "../..\..\..\include\novapi.h"

#define PRINTER_NAME L"novaPDF SDK 11"
#define PDF_TEXT L"Override options sample"
#define PDF_FILE_NAME L"OverrideOptions.pdf"

int _tmain(int argc, _TCHAR* argv[])
{
    HRESULT hr = S_OK;

    //initialize COM
    hr = CoInitialize(NULL);
    if (FAILED(hr))
    {
        MessageBox(NULL, L"Failed to initialize COM", L"novaPDF", MB_OK);
        return hr;
    }

    //create one NovaPdfOptions instance
    INovaPdfOptions11 *pNova = 0;
    hr = CoCreateInstance(__uuidof(NovaPdfOptions11), NULL, NULL,
                          CLSCTX_INPROC_SERVER, __uuidof(INovaPdfOptions11), (LPVOID*) &pNova);
    if (FAILED(hr))
    {
        MessageBox(NULL, L"Failed to create novaPDF COM object", L"novaPDF", MB_OK);
        return hr;
    }

    // initialize the NovaPdfOptions object to use with a printer licensed for SDK
    // if you have an application license for novaPDF SDK,
    // pass the license key to the Initialize() function
```
hr = pNova->Initialize(PRINTER_NAME, L""");

if (SUCCEEDED(hr)) {
    // Overide options from active profile
    // Save options
    // do not show prompt save dialog
    pNova->SetOverrideOptionLong(NOVPDF_OVERRIDE_SAVE_TYPE,
    PROMPT_SAVE_NONE);
    // save the pdf in next folder
    pNova->SetOverrideOptionLong(NOVPDF_OVERRIDE_SAVE_FOLDER_TYPE,
    SAVEFOLDER_CUSTOM);
    pNova->SetOverrideOptionString(NOVPDF_OVERRIDE_SAVE_FOLDER, L
    "C:\\temp");
    pNova->SetOverrideOptionString(NOVPDF_OVERRIDE_SAVE_FILE,
    PDF_FILE_NAME);
    // access folder, if restricted
    pNova->SetOverrideOptionEncryptedString(NOVPDF_OVERRIDE_SAVE_USER, L"user" );
    pNova->SetOverrideOptionEncryptedString(NOVPDF_OVERRIDE_SAVE_PASSWORD, L
    "password");
    // if a file with the same name already exists
    pNova->SetOverrideOptionLong(NOVPDF_OVERRIDE_SAVE_CONFLICT,
    FILE_CONFLICT_STRATEGY_OVERWRITE);
    // disable open PDF in viewer
    pNova->SetOverrideOptionBool(NOVPDF_OVERRIDE_OPEN_VIEWER,
    FALSE);
    // Embed fonts
    pNova->SetOverrideOptionBool(NOVPDF_OVERRIDE_EMBEDFONTS,
    TRUE);

    // Set compression level
    pNova->SetOverrideOptionLong(NOVPDF_OVERRIDE_COMPRESSION,
    COMPRESSION_HIGHQUALITY);
    // Merge PDF
    // Enable merge
    /*
     * pNova->SetOverrideOptionBool(NOVPDF_OVERRIDE_MERGE_PDF, TRUE);
     * // Merge with the same file, if the file exists in destination
     * OR merge with another file
     * pNova->SetOverrideOptionLong(NOVPDF_OVERRIDE_MERGE_FILE_TYPE,
     * MERGE_FILE_OTHER);
     * // Append to existing PDF, insert at the beginning or insert at
     * a page number
     * pNova->SetOverrideOptionLong(NOVPDF_OVERRIDE_MERGE_FILE_TYPE,
     * MERGE_TYPE_APPEND);
     * // if merge with other file, specify file name (and path)
     * p Nova-> SetOverrideOptionString(NOVPDF OVERRIDE_MERGE_FILE,
     * L"invoice-text.pdf");
     * // if merge type is set to insert at page number
     * pNova->SetOverrideOptionLong(NOVPDF_OVERRIDE_INSERT_PAGENO,
     * l);
    * // if the existing PDF is encrypted
// pNova->SetOverrideOptionEncryptedString(NOVAPDF_OVERRIDE/pdf_password, L"password");
   */

   //Document info
   pNova->SetOverrideOptionBool(NOVAPDF_OVERRIDE/doc_info, TRUE);
   pNova->SetOverrideOptionString(NOVAPDF_OVERRIDE/doc_title, L"sdk title");
   pNova->SetOverrideOptionString(NOVAPDF_OVERRIDE/doc_subject, L"sdk subject");
   pNova->SetOverrideOptionString(NOVAPDF_OVERRIDE/doc_author, L"sdk author");
   pNova->SetOverrideOptionString(NOVAPDF_OVERRIDE/doc_key, L"sdk key");
   pNova->SetOverrideOptionString(NOVAPDF_OVERRIDE/doc_creator, L"sdk creator");

   //Encrypt PDF
   /*
   pNova->SetOverrideOptionBool(NOVAPDF_OVERRIDE/security, TRUE);
   pNova->SetOverrideOptionEncryptedString(NOVAPDF_OVERRIDE/security_U, L"user");
   pNova->SetOverrideOptionEncryptedString(NOVAPDF_OVERRIDE/security_O, L"owner");
   */

   //Email
   pNova->SetOverrideOptionBool(NOVAPDF_OVERRIDE/send_email, TRUE);
   pNova->SetOverrideOptionString(NOVAPDF_OVERRIDE/rec_to, L"to");
   pNova->SetOverrideOptionString(NOVAPDF_OVERRIDE/rec_from, L"from");
   pNova->SetOverrideOptionString(NOVAPDF_OVERRIDE/rec_cc, L"cc");
   pNova->SetOverrideOptionString(NOVAPDF_OVERRIDE/rec_bcc, L"bcc");

   //register for events from novaPDF so we can wait until the
document is printed
   pNova->RegisterNovaEvent(NOVAPDF_EVENT_ACTIONS);

   HANDLE hPrinter;
   PDEVMODE pDevmode = NULL;
   PRINTER_DEFAULTS pd = { NULL, NULL, PRINTER_ACCESS_USE };

   //start print job
   if (OpenPrinter(PRINTER_NAME, &hPrinter, &pd)) {
      //get default printer DEVMODE
      int nSize = DocumentProperties(NULL, hPrinter, PRINTER_NAME, NULL, NULL, 0);
      pDevmode = (PDEVMODE)LocalAlloc(LPTR, nSize);
      DocumentProperties(NULL, hPrinter, PRINTER_NAME, pDevmode, NULL, DM_OUT_BUFFER);

      //set page size in DEVMODE
      pDevmode->dmPageSize = DMPAPER_USER;
Java

1.5.7.1 Hello World

Hello World (Java) sample is a basic Java console application that generates (using the novaPDF SDK 11 printer) one PDF file containing the text "Hello World from Java!". It demonstrates the basic use of the INovaPDFOptions interface with j-Interop.

What this sample does:
- determines the active profile, makes a copy of it and names it "Test Java"
- sets the new profile (Test Java) as active as well as some mandatory settings (e.g. the subject of the PDF)
- generates a test PDF file
- restores the original settings of the novaPDF SDK 11 printer driver.

"j-Interop is a Java Open Source library (under LGPL) that implements the DCOM wire protocol (MSRPC) to enable development of Pure, Bi-Directional, Non-Native Java applications which can interoperate with any COM component." j-Interop can be found at http://www.j-interop.org/

Note
If you encounter problems or have questions on using j-Interop, visit their FAQ page at http://www.j-interop.org/faq.html
Also, to avoid the “Logon failure: unknown user name or bad password” error, configure DCOM for remote access (detailed here - http://j-integra.intrinsyc.com/support/com/doc/remoteaccess.html#winxp) and make sure your firewall is not turned on.

SOURCE CODE OF THE HELLO WORLD JAVA SAMPLE (MAIN.JAVA):

```java
package helloworld;

import java.awt.*;
import java.awt.print.*;
import java.net.UnknownHostException;
import java.util.logging.Level;
import javax.print.PrintService;
import org.jinterop.dcom.common.JIException;
import org.jinterop.dcom.common.JISystem;
import org.jinterop.dcom.core.IJIComObject;
import org.jinterop.dcom.core.IJIComServer;
import org.jinterop.dcom.core.JIFragId;
import org.jinterop.dcom.core.JIProgId;
import org.jinterop.dcom.core.JISession;
import org.jinterop.dcom.core.JIString;
import org.jinterop.dcom.core.JIVariant;
import org.jinterop.dcom.impls.JIObjectFactory;
import org.jinterop.dcom.impls.autimation.IJIDispatch;

public class Main implements Printable {
    private static String PRINTER_NAME = "novaPDF SDK 11";
    private static String MESSAGE = "Hello world from Java2!";
    public static String PROFILE_NAME = "Test Java";
    public static int PROFILE_IS_PUBLIC = 0;
    public static long NOVAPDF_DOCINFO_SUBJECT = 68;
    public static String NOVAPDF_INFO_SUBJECT = "Java Hello World Document Subject";
    public static long NOVAPDF_SAVE_FILE_NAME = 104;
    public static long NOVAPDF_SAVE_FILEEXIST_ACTION = 108;
```
public static long FILE_CONFLICT_STRATEGY_AUTONUMBER_NEW = 1;
public static long NOVAPDF_INFO_VIEWER_ENABLE = 8;
public static long NOVAPDF_SAVE_PROMPT_TYPE = 101;
public static long PROMPT_SAVE_SIMPLE = 2;

JIComServer comStub = null;
IJIDispatch pNovaDispatch = null;
IJIComObject pNova = null;

/**
 * @param args the command line arguments
 */
public static void main(String[] args) throws JIException, UnknownHostException {
    if (args.length < 4) {
        System.out.println("Please provide address domain username password");
        return;
    }
    new Main().doPrint(args);
}

public void doPrint(String[] args) throws JIException, UnknownHostException {

    //disable J-Interop Log
    try {
        JISystem.getLogger().setLevel(Level.INFO);
        JISystem.setInBuiltLogHandler(false);
    } catch (Exception e) {
        //System.out.println(e.getMessage());
    }

    //connecting to COM/DCOM server
    JISession session = JISession.createSession(args[1], args[2], args[3]);
    session.useSessionSecurity(true);

    JIProgId pid = JIProgId.valueOf("novapi11.NovaPdfOptions11");
    pid.setAutoRegistration(true);

    comStub = new JIComServer(pid, args[0], session);
    pNova = comStub.createInstance();

    pNovaDispatch = (IJIDispatch) JIObjectFactory.narrowObject(pNova.
queryInterface(IJIDispatch.IID));
    JIString PRINTER = new JIString(PRINTER_NAME);

    try {
        pNovaDispatch.callMethodA("Initialize2", new Object[]{PRINTER},
new JIString(""));
        pNovaDispatch.callMethodA("LicenseShellExecuteFile", new Object
[]{new JIString("java")});
    } catch (JIException except) {
        System.out.println("Initialize/LicenseShellExecuteFile");
        except.printStackTrace();
    }
String oldActiveProfile = "";
try {
    JIVariant ap[] = pNovaDispatch.callMethodA("GetActiveProfile2",
    new Object[]{new JIVariant("", true)});
    oldActiveProfile = ap[1].getObjectAsString().getString();
} catch (JIException except) {
    System.out.println("GetActiveProfile2");
    except.printStackTrace();
}

//ADD A NEW PROFILE
String activeProfile = "";
try {
    JIVariant ap[] = pNovaDispatch.callMethodA("AddProfile2",
    new Object[]{new JIString(PROFILE_NAME), PROFILE_IS_PUBLIC, new JIVariant("", true)});
    activeProfile = ap[1].getObjectAsString().getString();
} catch (JIException except) {
    System.out.println("AddProfile2");
    except.printStackTrace();
}

//LOAD PROFILE
try {
    pNovaDispatch.callMethod("LoadProfile2", new Object[]{new JIString(activeProfile)});
} catch (JIException except) {
    System.out.println("LoadProfile2");
    except.printStackTrace();
}

//CHANGE SOME OPTIONS
try {
    // and set some options
    pNovaDispatch.callMethod("SetOptionString2", new Object[]{
        NOVAPDF_DOCINFO_SUBJECT, new JIString(NOVAPDF_INFO_SUBJECT)
    });
    pNovaDispatch.callMethod("SetOptionString2", new Object[]{
        NOVAPDF_SAVE_FILE_NAME, new JIString("novaPDFJavaDocument")
    });
    pNovaDispatch.callMethod("SetOptionLong", new Object[]{
        NOVAPDF_SAVE_FILEEXIST_ACTION, FILE_CONFLICT_STRATEGY_AUTONUMBER_NEW
    });
    pNovaDispatch.callMethod("SetOptionBool", new Object[]{
        NOVAPDF_INFO_VIEWER_ENABLE, 1
    });
    pNovaDispatch.callMethod("SetOptionLong", new Object[]{
        NOVAPDF_SAVE_PROMPT_TYPE, PROMPT_SAVE_SIMPLE
    });
} catch (JIException except) {
    System.out.println("Set options");
    except.printStackTrace();
}

//SAVE PROFILE
try {
    pNovaDispatch.callMethod("SaveProfile2");
} catch (JIException except) {
    System.out.println("SaveProfile2");
    except.printStackTrace();
}

//SET IT AS ACTIVE PROFILE
try {
    // set the copy profile as active profile ...
    pNovaDispatch.callMethod("SetActiveProfile2", new Object[]{new
    JIString(activeProfile)});
} catch (JIException except) {
    System.out.println("SetActiveProfile2");
    except.printStackTrace();
}

//PRINT SOMETHING WITH NEW PROFILE
PrinterJob job = PrinterJob.getPrinterJob();
PrintService[] services = PrinterJob.lookupPrintServices();

Boolean flag = Boolean.FALSE;
for (int i = 0; i < services.length; i++) {
    if (services[i].getName().equalsIgnoreCase(PRINTER_NAME)) {
        flag = Boolean.TRUE;
        try {
            job.setPrintService(services[i]);
            job.setPrintable(this);
            job.print();
        } catch (Throwable throwable) {
            System.out.println("Printing Exception");
            throwable.printStackTrace();
        }
        break;
    }
}

if(flag == Boolean.FALSE){
    System.out.println("Printer not found...");
} else {
    System.out.println("PDF Printed");
}

//RESTORE PREVIOUS ACTIVE PROFILE
try {
    if(oldActiveProfile.length() > 0) {
        pNovaDispatch.callMethod("SetActiveProfile2", new Object[]{
        new JIString(oldActiveProfile)});
    }
} catch (JIException except) {
    System.out.println("SetActiveProfile2 Old");
    except.printStackTrace();
}

//DELETE THE NEW PROFILE
try {
    pNovaDispatch.callMethod("DeleteProfile2", new Object[]{
    new JIString(activeProfile)});
}
catch (JIException except) {
    System.out.println("DeleteProfile2");
    except.printStackTrace();
}

JISession.destroySession(session);

public int print(Graphics g, PageFormat pf, int page) throws PrinterException {
    if (page > 0) { /* We have only one page, and 'page' is zero-based */
        return NO_SUCH_PAGE;
    }
    /* User (0,0) is typically outside the imageable area, so we must translate by the X and Y values in the PageFormat to avoid clipping */
    Graphics2D g2d = (Graphics2D) g;
    g2d.translate(pf.getImageableX(), pf.getImageableY());
    /* Now we perform our rendering */
    g.drawString(MESSAGE, 100, 100);
    /* tell the caller that this page is part of the printed document */
    return PAGE_EXISTS;
}

1.5.7.2 Word OLE

The Word OLE (Java) SDK sample is a basic Java console application that converts a MS Word document (in this sample the default location for the source document is C:\temp\test.doc) to PDF using Word OLE automation and j-Interop.

"j-Interop is a Java Open Source library (under LGPL) that implements the DCOM wire protocol (MSRPC) to enable development of Pure, Bi-Directional, Non-Native Java applications which can interoperate with any COM component."

Note
If you encounter problems or have questions on using j-Interop, visit their FAQ page at http://www.j-interop.org/faq.html
Also, to avoid the “Logon failure: unknown user name or bad password” error, configure DCOM for remote access (detailed here - http://j-integra.intrinsyc.com/support/com/doc/remoteaccess.html#winxp) and make sure your firewall is not turned on.

SOURCE CODE FOR WORD OLE SAMPLE (MAIN.JAVA):

```java
package wordole;

import java.io.File;
```
```java
import java.util.logging.Level;
import org.jinterop.dcom.common.JIException;
import org.jinterop.dcom.common.JISystem;
import org.jinterop.dcom.core.IJICOMOBJECT;
import org.jinterop.dcom.core.JICOMSERVER;
import org.jinterop.dcom.core.JIPROGID;
import org.jinterop.dcom.core.JISESSION;
import org.jinterop.dcom.core.JISTRING;
import org.jinterop.dcom.core.JIVARIANT;
import org.jinterop.dcom.impls.JIObjectFactory;
import org.jinterop.dcom.impls.automation.IJIDispatch;

public class Main {
    public static String PROFILE_NAME = "Test Java and Word OLE";
    public static int PROFILE_IS_PUBLIC = 0;
    public static long NOVAPDF_DOCINFO_SUBJECT = 68;
    public static String NOVAPDF_INFO_SUBJECT = "Java OLE Word Document Subject";
    public static long NOVAPDF_SAVE_FILE_NAME = 104;
    public static long NOVAPDF_SAVE_FILEEXIST_ACTION = 108;
    public static long FILE_CONFLICT_STRATEGY_AUTONUMBER_NEW = 1;
    public static long NOVAPDF_INFO_VIEWER_ENABLE = 8;
    public static long NOVAPDF_SAVE_PROMPT_TYPE = 101;
    public static long PROMPT_SAVE_SIMPLE = 2;

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        if (args.length < 4) {
            System.out.println("Please provide address domain username password");
            return;
        }
        File docFile = new File("c:\temp\test.doc");
        if (!docFile.exists()) {
            System.out.println("c:\temp\test.doc file does not exist");
            return;
        }
        //disable J-Interop Log
        try {
            JISystem.getLogger().setLevel(Level.INFO);
            JISystem.setBuiltInLogHandler(false);
        } catch (Exception e) {
            //System.out.println(e.getMessage());
        }
        JICOMSERVER comStub = null;
        IJIDispatch pNovaDispatch = null;
        IJICOMOBJECT unknown = null;
        try {
            JISESSION session = JISESSION.createSession(args[1], args[2],
```

arguments[3]);
    session.useSessionSecurity(true);

    JIProgId pid = JIProgId.valueOf("novapi11.NovaPdfOptions11");
    pid.setAutoRegistration(true);

    comStub = new JIComServer(pid, arguments[0], session);
    unknown = comStub.createInstance();

    pNovaDispatch = (IJIDispatch) JIObjectFactory.narrowObject(
                     unknown queryInterface(IJIDispatch.IID));

    // INITIALIZE
    JIString PRINTER = new JIString("novaPDF SDK 11");
    JIString APPNID = new JIString("Word.Application");
    pNovaDispatch.callMethod("Initialize2", new Object[]{PRINTER},
                           new JIString(""));

    // GET ACTIVE PROFILE
    String oldActiveProfile = "";
    try {
        JIVariant ap[] = pNovaDispatch.callMethodA(  
            "GetActiveProfile2", new Object[]{new JIVariant("", true)});
        oldActiveProfile = ap[1].getObjectAsString().getString();
    } catch (JIException except) {
        System.out.println("GetActiveProfile2");
        except.printStackTrace();
    }

    String activeProfile = "";
    // ADD A NEW PROFILE
    try {
        JIVariant ap[] = pNovaDispatch.callMethodA("AddProfile2",  
            new Object[]{new JIString(PROFILE_NAME), PROFILE_IS_PUBLIC, new JIVariant("", true)});
        activeProfile = ap[1].getObjectAsString().getString();
    } catch (JIException except) {
        System.out.println("AddProfile2");
        except.printStackTrace();
    }

    // LOAD PROFILE
    try {
        pNovaDispatch.callMethod("LoadProfile2", new Object[]{new
                        JIString(activeProfile)});
    } catch (JIException except) {
        System.out.println("LoadProfile2");
        except.printStackTrace();
    }

    // CHANGE SOME OPTIONS
    try {
        // and set some options
        pNovaDispatch.callMethod("SetOptionString2", new Object[]{

NOVAPDF_DOCINFO_SUBJECT, new JIString(NOVAPDF_INFO_SUBJECT));
pNovaDispatch.callMethod("SetOptionString2", new Object[][][{.
NOVAPDF_SAVE_FILE_NAME, new JIString("novaPDFJavaDocument"));
pNovaDispatch.callMethod("SetOptionLong", new Object[][][{.
NOVAPDF_SAVE_FILEEXIST_ACTION, FILE_CONFLICT_STRATEGY_AUTONUMBER_NEW});
pNovaDispatch.callMethod("SetOptionBool", new Object[][][{.
NOVAPDF_INFO_VIEWER_ENABLE, 1});
pNovaDispatch.callMethod("SetOptionLong", new Object[][][{.
NOVAPDF_SAVE_PROMPT_TYPE, PROMPT_SAVE_SIMPLE});
} catch (JISEception except) { 
  System.out.println("Set options");
  except.printStackTrace();
}

//SAVE PROFILE
try {
  pNovaDispatch.callMethod("SaveProfile2");
} catch (JISEception except) {
  System.out.println("SaveProfile2");
  except.printStackTrace();
}

//SET IT AS ACTIVE PROFILE
try {
  // set the copy profile as active profile ... 
  pNovaDispatch.callMethod("SetActiveProfile2", new Object[][][{.
      new JIString(activeProfile)));
  } catch (JISEception except) {
    System.out.println("SetActiveProfile2");
    except.printStackTrace();
  }

//InitializeOLEUsage("Word.Application");
pNovaDispatch.callMethod("InitializeOLEUsage", new Object[][{.
    APPNID));
pNovaDispatch.callMethod("LicenseOLEServer");

MSWord word = new MSWord(args[0], args, pNovaDispatch, PRINTER);

word.startWord();
word.showWord();

word.performOp();
word.quitAndDestroy();

//RESTORE PREVIOUS ACTIVE PROFILE
try {
  if (oldActiveProfile.length() > 0) { 
    pNovaDispatch.callMethod("SetActiveProfile2", new Object[][{new JIString(oldActiveProfile)));
  }
} catch (JISEception except) {
  System.out.println("SetActiveProfile2 Old");
  except.printStackTrace();
}

//DELETE THE NEW PROFILE
try {
    pNovaDispatch.callMethod("DeleteProfile2", new Object[]{new JIString(activeProfile)});
    } catch (JIException except) {
        System.out.println("DeleteProfile2");
        except.printStackTrace();
    } System.out.println("Done!");

    } catch (Exception e) {
        System.out.println("---UPS---");
        e.printStackTrace();
    }
}

SOURCE CODE FOR WORD OLE SAMPLE (MSWORD.JAVA):

```java
package wordole;
/*
 * j-interop can be found at http://www.j-interop.org/
 * and it is an open source project
 *
 */
import java.net.UnknownHostException;
import org.jinterop.dcom.common.JIException;
import org.jinterop.dcom.common.JISystem;
import org.jinterop.dcom.core.IJIComObject;
import org.jinterop.dcom.core.JIComServer;
import org.jinterop.dcom.core.JIProgId;
import org.jinterop.dcom.core.JISession;
import org.jinterop.dcom.core.JIString;
import org.jinterop.dcom.core.JIVariant;
import org.jinterop.dcom.impls.JIObjectFactory;
import org.jinterop.dcom.impls.automation.IJIDispatch;

public class MSWord {
    private JIComServer comStub = null;
    private IJIDispatch dispatch = null;
    private IJIComObject unknown = null;
    private IJIDispatch PDF = null;
    private JIString PRINTER = null;

    public MSWord(String address, String[] args, IJIDispatch PDF, JIString PRINTER) throws JIException, UnknownHostException {
        this.PDF = PDF;
        this.PRINTER = PRINTER;

        JISession session = JISession.createSession(args[1], args[2], args[3]);
        session.useSessionSecurity(true);
    }
```
comStub = new JIComServer(JIProgId.valueOf("Word.Application"),
address, session);
}

public void startWord() throws JIException {
    unknown = comStub.CreateInstance();
    dispatch = (IJIDispatch) JIOBJECTFACTORY.narrowObject(unknown.
    queryInterface(IJIDispatch.IID));
}

public void showWord() throws JIException {
    int dispId = dispatch.getIDsOfNames("Visible");
    JIVariant variant = new JIVariant(Boolean.FALSE);
    dispatch.put(dispId, variant);
    PDF.callMethod("LicenseOLEServer");
}

public void performOp() throws JIException, InterruptedException {
    /* JISystem config */
    /*
     */
    JISystem.setJavaCoClassAutoCollection(true);
    System.out.println(((JIVariant) dispatch.get("Version")).
    getString());
    System.out.println(((JIVariant) dispatch.get("Path")).
    getString());
    JIVariant variant = dispatch.get("Documents");
    System.out.println("Open document...");
    IJIDispatch documents = (IJIDispatch) JIOBJECTFACTORY.
    narrowObject(variant.getobjectAsComObject());
    JIString filePath = new JIString("c:\temp\test.doc");
    JIVariant variant2[] = documents.callMethodA("open", new Object []
    { filePath, JIVariant.OPTIONAL_PARAM(), JIVariant.OPTIONAL_PARAM(),
    JIVariant.OPTIONAL_PARAM(), JIVariant.OPTIONAL_PARAM(), JIVariant.
    OPTIONAL_PARAM(), JIVariant.OPTIONAL_PARAM(), JIVariant.OPTIONAL_PARAM() ,
    JIVariant.OPTIONAL_PARAM(), JIVariant.OPTIONAL_PARAM(), JIVariant.
    OPTIONAL_PARAM(), JIVariant.OPTIONAL_PARAM(), JIVariant.OPTIONAL_PARAM() ,
    JIVariant.OPTIONAL_PARAM() });
    System.out.println("doc opened");
    //ActivePrinter
    dispatch.put("ActivePrinter", new Object[]{PRINTER});
    sleep(5);
    System.out.println("Get content...");
    IJIDispatch document = (IJIDispatch) JIOBJECTFACTORY.
    narrowObject(variant2[0].getobjectAsComObject());
    sleep(5);
    System.out.println("Printing...");
    document.callMethod("PrintOut", new Object[] {1});
    System.out.println("Closing document...");
private void sleep(int sec) throws InterruptedException {
    System.out.println("Sleeping "+sec+" second(s)... ");
    Thread.sleep((int)(sec * 1000));
}

/**
 * @throws JIException
 */
public void quitAndDestroy() throws JIException {
    System.out.println("Quit...");
    dispatch.callMethod("Quit", new Object[]{new JIVariant(-1,
        true), JIVariant.OPTIONAL_PARAM(), JIVariant.OPTIONAL_PARAM() });
    JISession.destroySession(dispatch.getAssociatedSession());
}

1.5.8 VBNNet

1.5.8.1 Hello World VBNNet

Hello World VBNNet sample is a simple Windows console application that prints one page with the "novaPDF says Hello World ..." text to the novaPDF SDK 11.

It demonstrates the basic use of the INovaPDFOptions interface. The printing job is made using the package "System.Drawing.Printing"

Basically the sample determines the active profile, makes a copy of it into a profile called "Test VBNNet", sets the new profile as active, sets the subject of the generated PDF document, prints a page, and restores original printer settings. The location of the generated document depends on whatever the settings are for the current active profile.

Notice

Because of the specific exception based error handling in .NET, all calls to methods in the INovaPDFOptions interface must be nested within a try-catch block. Consider for example that we want to add a profile called "test", but the profile "test" already exists. Then the call pNova.AddProfile("test") will throw an "System.Runtime.InteropServices.COMException". with the ErrorCode property set to NV_PROFILE_EXISTS (0xD5DA0006).

Source code

Imports System
Imports System.Drawing
Imports System.Drawing.Printing
Imports System.Windows.Forms
' the novapiLib package must be added as a COM reference
Imports novapiLib80
Module Module1
    ' <summary>
    ' The main entry point for the application.
    ' </summary>
    Const PRINTER_NAME As String = "novaPDF SDK 11"

    Const PROFILE_NAME As String = "Test VBNet"
    Const PROFILE_IS_PUBLIC As Integer = 0
    Const NOVAPDF_INFO_SUBJECT As String = "Document Subject"

    Const NV_PROFILE_EXISTS As Long = -707133434

    Sub Main()
        Try
            ' create the NovaPdfOptions object
            Dim pNova As NovaPdfOptions11
            pNova = New NovaPdfOptions11
            ' initialize the NovaPdfOptions object
            ' if you have an application license for novaPDF SDK,
            pNova.Initialize(PRINTER_NAME)
            ' mark start changing options
            ' get the active profile ...
            Dim activeProfile As String = ""
            Dim nActivePublic As Integer = 0
            Try
                pNova.GetActiveProfile(activeProfile)
                Catch ex As System.Runtime.InteropServices.COMException
                    ' ignore profile exists error
                    If (NovaErrors.NV_NO_ACTIVE_PROFILE = ex.ErrorCode) Then
                        System.Console.WriteLine("The printer does not have an
                        active profile")
                    Else
                        ' more serious error, propagate it
                        Throw ex
                    End If
                End Try

                Dim newProfileID As String = ""
                Try
                    ' and make a copy of it
                    pNova.AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC,
                                    newProfileID)
                    Catch e As System.Runtime.InteropServices.COMException
                        ' ignore profile exists error
                        If (NV_PROFILE_EXISTS = e.ErrorCode) Then
                            System.Console.WriteLine("Profile already
                            exists")
                        Else
                            ' more serious error, propagate it
                            Throw e
                        End If
                    End Try

                    ' load the new profile
                    pNova.LoadProfile(newProfileID)
                    'nova.NovaTools.AddProfileOptions(pNova);

                    ' and set some options
                    ' uncomment the function calls for the options you wish to set
                    ' change the options in the nova.cs unit
'nova.NovaTools.AddProfileOptions(pNova);
'nova.NovaTools.AddDocumentInformation(pNova);
'nova.NovaTools.AddViewerOptions(pNova);
'nova.NovaTools.AddLinksOptions(pNova);
'nova.NovaTools.AddAdvancedOptions(pNova);
'nova.NovaTools.AddGraphicsOptions(pNova);
'nova.NovaTools.AddSecurityOptions(pNova);
'nova.NovaTools.AddSaveOptions(pNova);
'nova.NovaTools.AddAfterSaveActions(pNova);
'nova.NovaTools.AddMailOptions(pNova);
'nova.NovaTools.AddWatermarkImage(pNova);
'nova.NovaTools.AddPageText(pNova);
'nova.NovaTools.AddOverlayOptions(pNova);
'nova.NovaTools.AddSignatureOptions(pNova);
'nova.NovaTools.AddEmbedFontsOptions(pNova);
'nova.NovaTools.AddBookmarksDefinitions(pNova);

'save the new added profile
pNova.SaveProfile()
' set the copy profile as active profile ...
pNova.SetActiveProfile(newProfileID)

' print a test page, using the previously set active profile settings
Dim pd As PrintDocument = New PrintDocument
pd.PrinterSettings.PrinterName = PRINTER_NAME
AddHandler pd.PrintPage, AddressOf PrintPageFunction
pd.Print()
If ((activeProfile.Length > 0) And
(activeProfile.CompareTo(PROFILE_NAME) <> 0)) Then
    pNova.SetActiveProfile(activeProfile)
    pNova.DeleteProfile(newProfileID)
End If
Catch e As System.Runtime.InteropServices.COMException
    MessageBox.Show(e.Message)
Catch e As Exception
    MessageBox.Show(e.Message)
End Try
End Sub

' and finally the function that actually prints the page
Private Sub PrintPageFunction(ByVal sender As Object, ByVal ev As PrintPageEventArgs)
Dim str As String = "novaPDF says Hello World from VB.Net"
Dim font As Font = New Font("Arial", 16)
Dim brush As Brush = New SolidBrush(Color.Black)
ev.Graphics.DrawString(str, font, brush, 20.0!, 20.0!)
ev.HasMorePages = False
End Sub

End Module

1.5.8.2 VBNet Converter

The VBNet Converter sample demonstrates how to convert an existing file by printing it to novaPDF SDK 11 using the ShellExecute function. It also demonstrates how to set different options and manage profiles.
The same approach should be taken if you print using a "Print()" method from another object (like an internet browser or a report control). Just replace the ShellExecute call with the call of your Print method.

When the application starts, it creates a few profiles and makes different settings in the profiles. Then it shows a dialog from where the user can select the active profile and change its settings using the controls from the dialog.

After that a document can be selected from the harddisk and printed to novaPDF SDK 11 using the ShellExecute function call.

When using this technique to convert a file to PDF, you have to take care of the fact that ShellExecute prints to the default printer. This function returns immediately and does not wait until the print is finished (it may return before the printing is actually started). Therefore you have to set the default printer to novaPDF SDK 11 before calling ShellExecute (using the SetDefaultPrinter method), wait the process to be started (using WaitForExit()), restore the default printer (with the RestoreDefaultPrinter method). This way you made sure that the default printer was restored and your document is printed to novaPDF SDK 11.

Source code snippets

1. Declare INovaPdfOptions variable
   ```vb
   Private mobjNovaOptions As NovaPdfOptions80Class
   ```

2. Initialize INovaPdfOptions
   ```vb
   mobjNovaOptions = New NovaPdfOptions80Class
   ' initialize the NovaPdfOptions object
   mobjNovaOptions.Initialize(PRINTER_NAME)
   AddSmallProfile()
   AddFullProfile()
   ```

3. Set novaPDF SDK 11 Options
   ```vb
   Try
   Dim newSmallSizeProfileId As String = ""
   ' add new profile
   mobjNovaOptions.AddProfile(SMALL_SIZE_PROFILE, PROFILE_IS_PUBLIC, newSmallSizeProfileId)
   ' load the new profile
   mobjNovaOptions.LoadProfile(newSmallSizeProfileId)
   ' Set some options to this profile
   ' disable the "Save PDF file as" prompt
   mobjNovaOptions.SetOptionLong(NovaOptions.NOVPDF_SAVE_PROMPT_TYPE, 0)
   ' set generated Pdf files destination folder "c:\"
   mobjNovaOptions.SetOptionString(NovaOptions.NOVPDF_SAVE_FOLDER, "c:\")
   // .....  
   Catch ComException As System.Runtime.InteropServices.COMException
   ```
4. Start a Print job

Private Sub btnStartPrinting_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnStartPrinting.Click
    mobjNovaOptios.SetActiveProfile(CType((cmbProfiles.SelectedItem), String))
    mobjNovaOptios.SetDefaultPrinter()
    mobjNovaOptios_LICENSESHULLEXECUTEFILE(TxtFileToConvert.Text)
    Dim myProcess As Process = New Process
    Try
        myProcess.StartInfo.FileName = txtFileToConvert.Text
        myProcess.StartInfoVerb = "Print"
        myProcess.StartInfo.CreateNoWindow = True
        myProcess.Start()
        Catch ex As Win32Exception
            If ex.NativeErrorCode = ERROR_FILE_NOT_FOUND Then
                Console.WriteLine(ex.Message + ". Check the path and filename")
            Else
                ' Note that if your word processor might generate exceptions such as this, which are handled first.
                If ex.NativeError = ERROR_ACCESS_DENIED Then
                    Console.WriteLine(ex.Message + ". You do not have permission to print this file.")
                End If
            End If
        End Try
        myProcess.WaitForExit(10000)
        myProcess.Close()
        mobjNovaOptios.RestoreDefaultPrinter()
    End Sub

1.5.8.3 Word OLE VBN

The Word OLE VBN sample is a simple Windows console application that converts a MS Word document (C:\Test.doc) to PDF using Word OLE automation.

Source code

Imports System
Imports System.Drawing
Imports System.Drawing.Printing
Imports System.Windows.Forms
Imports novapiLib80

Module Module1
    ' <summary>
    ' The main entry point for the application.
    '</summary>
Const PRINTER_NAME As String = "novaPDF SDK 11"
Const PROFILE_NAME As String = "Word OLE VBNNet"
Const PROFILE_IS_PUBLIC As Integer = 0
Const NOVAPDF_INFO_SUBJECT As Integer = 68

Sub Main()

' create the NovaPdfOptions80 object
Dim pNova As NovaPdfOptions11
pNova = New NovaPdfOptions11
Dim activeProfile As String = ""
Dim newProfileID As String = String.Empty

' initialize the NovaPdfOptions110 object
' if you have an application license for novaPDF SDK,
pNova.Initialize(PRINTER_NAME, "")
' save old active profile id
pNova.SetActiveProfile(activeProfile)
' create new profile
pNova.AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC, newProfileID)
' load the new profile
pNova.LoadProfile(newProfileID)
' and set some options
pNova.SetOptionString(NOVAPDF_INFO_SUBJECT, "C# Word OLE document")
pNova.SaveProfile()

' set the copy profile as active profile ...
pNova.SetActiveProfile(newProfileID)
' set nova default printer
pNova.SetDefaultPrinter()
' print word document
Dim objWord As Object
Dim objDoc As Object
pNova.InitializeOLEUsage("Word.Application")
objWord = CreateObject("Word.Application")
pNova_LICENSEOLESERVER()
objDoc = objWord.Documents.Open("C:\temp\test.docx", False, True)
objDoc.PrintOut(False)
objDoc.Close(False)
objWord.Quit(False)

' restore active profile
If activeProfile.Length > 0 AndAlso activeProfile.CompareTo(PROFILE_NAME) <> 0 Then
    pNova.SetActiveProfile(activeProfile)
pNova.DeleteProfile(newProfileID)
End If

' restore default printer
pNova.RestoreDefaultPrinter()
End Sub
End Module

1.5.8.4 Convert Office Docs

The Convert Office Docs sample is a simple Windows console application that converts some Microsoft Office documents (Word, Excel, Publisher, PowerPoint and Visio) using Convert... functions. It uses Office OLE automation so Microsoft Office has to be installed.
The sample documents converted are located in the "OfficeDocuments" folder in novaPDF SDK installation folder. The pdf files are saved in the same folder.

Hidden links and internal document links are converted in pdf hyperlinks (except for Excel documents).

Source code

```vbnet
Imports System
Imports System.Windows.Forms
' the novapLib package must be added as a COM reference
Imports novapLib1

Module Module1
  ' <summary>
  ' The main entry point for the application.
  ' </summary>
Const PRINTER_NAME As String = "novaPDF SDK 11"
Const PROFILE_NAME As String = "ConvertOffice"
Const PROFILE_IS_PUBLIC As Integer = 0
Const NV_PROFILE_EXISTS As Long = -707133434

Sub Main()
  Try
    ' create the NovaPdfOptions object
    Dim pNova As NovaPdfOptions11
    pNova = New NovaPdfOptions11
    ' initialize the NovaPdfOptions object
    ' if you have an application license for novaPDF SDK,
    ' pass the license key to the Initialize() function
    pNova.Initialize(PRINTER_NAME,"")
    ' mark start changing options
    ' get the active profile ...
    Dim activeProfile As String ="
    Dim nActivePublic As Integer = 0
    Try
      pNova.GetActiveProfile(activeProfile)
      Catch ex As System.Runtime.InteropServices.COMException
        ' ignore profile exists error
        If (NovaErrors.NV_NO_ACTIVE_PROFILE = ex.ErrorCode) Then
          System.Console.WriteLine("The printer does not have an active profile")
        Else
          ' more serious error, propagate it
          Throw ex
        End If
    End Try
    Dim newProfileID As String ="
    Try
      ' and make a copy of it
      pNova.AddProfile(PROFILE_NAME, PROFILE_IS_PUBLIC, newProfileID)
    Catch e As System.Runtime.InteropServices.COMException
      ' ignore profile exists error
      If (NV_PROFILE_EXISTS = e.ErrorCode) Then
        System.Console.WriteLine("Profile already exists")
    End Try
  End Try
End Sub
```
Else
    ' more serious error, propagate it
    Throw e
End If
End Try

'load the new profile
pNova.LoadProfile(newProfileID)
'nova.NovaTools.AddProfileOptions(pNova);

' and set some options
'uncomment the function calls for the options you wish to set

' DO NOT OPEN PDF
' an Open action is added by default in the profile - delete or disable this open action
' pNova.DisableActionType(ActionType.NOVA_ACTION_OPEN)

' change the save options
NovaTools.AddSaveOptions(pNova)

'save the new added profile
pNova.SaveProfile()
'set the copy profile as active profile ...
pNova.SetActiveProfile(newProfileID)

'Path for novaPDF 11 SDK sample documents
Dim strDocumentsPath As String = "C:\Users\Public\Documents\novaPDF 11\SDK\OfficeDocuments\"
Dim strDocPath As String

' --------------------------------------------------

' Convert Word document
'Microsoft Office Word has to be installed
' --------------------------------------------------

'Document file path
strDocPath = strDocumentsPath + "Word.docx"
'style name  | level  | type
Dim strHeadings As String = "Heading 1|1|0;Heading 2|2|0;Heading 3|3|0"
pNova.LicenseApplication("WINWORD.EXE")
pNova.LicenseApplication("SPLNOW64.EXE")
pNova.ConvertWordDocument(strDocPath, 1, 0, 1, 1, 1, 1, 1, 3, strHeadings)

' --------------------------------------------------

' Convert PowerPoint document
'Microsoft Office PowerPoint has to be installed
' --------------------------------------------------

'Document file path
strDocPath = strDocumentsPath + "PowerPoint.pptx"
pNova.LicenseApplication("POWERPNT.EXE")
pNova.LicenseApplication("SPLNOW64.EXE")
pNova.ConvertPowerPointDocument(strDocPath, 1, 1, 1, 1, 1, 1)

' --------------------------------------------------

' Convert Publisher document
'Microsoft Office Publisher has to be installed
' --------------------------------------------------
'Document file path
strDocPath = strDocumentsPath + "Publisher.pub"
pNova.LicenseApplication("MSPUB.EXE")
pNova.LicenseApplication("SPLWOW64.EXE")
pNova.ConvertPublisherDocument(strDocPath, 1, 1, 1, 1, 1)

' Convert Excel document
' Microsoft Office Excel has to be installed
' hidden links in Excel documents will not be converted to pdf links
'Document file path
strDocPath = strDocumentsPath + "Excel.xlsx"
pNova.LicenseApplication("EXCEL.EXE")
pNova.LicenseApplication("SPLWOW64.EXE")

' Convert Visio document
' Microsoft Office Visio has to be installed
'Document file path
strDocPath = strDocumentsPath + "Visio.vsd"
pNova.LicenseApplication("VISIO.EXE")
pNova.LicenseApplication("SPLWOW64.EXE")
pNova.ConvertVisioDocument(strDocPath, 1, 1, 1, 1, 1)

pNexta.SetActiveProfile(activeProfile)
pNova.DeleteProfile(newProfileID)
Catch e As System.Runtime.InteropServices.COMException
    MessageBox.Show(e.Message)
Catch e As Exception
    MessageBox.Show(e.Message)
End Try
End Sub
End Module

1.5.8.5 Override Options

Override Options sample is a simple Windows console application that demonstrates the basic use of the INovaPDFOptions interface, without using profiles options. Instead of creating a profile and setting options in the profile, the sample uses SetOverrideOptions... functions to quickly set some options that will overwrite active profile settings. This is useful if you have a simple application where you just wish to change the pdf file name, email address or other simple options for each printed document.

There is a limited set of options that can be set with SetOverrideOptions... functions, usually settings that users can also change from the Save As dialog or the other prompt dialogs that can be shown before printing a document. If you need more complex settings, like adding a watermark, overlay or signature you have to configure them in a profile.

You can combine usage of profiles and override options functions. For instance you can create a profile with a watermark that is set as active and then use the override function just to change the file name for each document.
Once set, override options are taken in account for all printed documents until they are removed with DeleteOverrideOptions function.

The sample prints one page with a simple text to the novaPDF SDK 11 and generates a "OverrideOptions.pdf" file in the C:\temp folder.

Source code

```csharp
Imports System
Imports System.Drawing
Imports System.Drawing.Printing
Imports System.Windows.Forms
' the novapilib package must be added as a COM reference
Imports novapilib11

Module Module1
' <summary>
' The main entry point for the application.
' </summary>
Const PRINTER_NAME As String = "novaPDF SDK 11"
Const PDF_FILE_NAME As String = "OverrideOptions.pdf"

Sub Main()
    Try
        ' create the NovaPdfOptions object
        Dim pNova As NovaPdfOptions11
        pNova = New NovaPdfOptions11

        ' initialize the NovaPdfOptions object
        ' if you have an application license for novaPDF SDK,
        ' pass the license key to the Initialize() function
        pNova.Initialize(PRINTER_NAME, "")

        'Override options from active profile

        'Save options
        'do not show prompt save dialog
        pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_SAVE_TYPE, SaveDlgType.PROMPT_SAVE_NONE)
        'save the pdf in next folder

        pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_SAVE_FOLDER_TYPE, SaveFolder.SAVEFOLDER_CUSTOM)

        pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_SAVE_FOLDER, "C:\temp")

        pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_SAVE_FILE, PDF_FILE_NAME)
        'access folder, if restricted

        pNova.SetOverrideOptionEncryptedString(NovaOptions.NOVAPDF_OVERRIDE_SAVE_USER, "user")

        pNova.SetOverrideOptionEncryptedString(NovaOptions.NOVAPDF_OVERRIDE_SAVE_PASSWORD, "password")
    End Try
End Sub
```
'if a file with the same name already exists

pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_SAVE_CONFLICT, SaveFileConflictType.FILE_CONFLICT_STRATEGY_OVERWRITE)

'disable open PDF in viewer

'pNova.SetOverrideOptionBool(NovaOptions.NOVAPDF_OVERRIDE_OPEN_VIEWER, 0)

'Embed fonts

pNova.SetOverrideOptionBool(NovaOptions.NOVAPDF_OVERRIDE_EMBEDFONTS, 1)

'Set compression level

'pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_COMPRESSION, CompressionOverride.COMPRESSION_HIGHQUALITY)

'Merge PDF

'Enable merge

'pNova.SetOverrideOptionBool(NovaOptions.NOVAPDF_OVERRIDE_MERGE_PDF, 1)

'Merge with the same file, if the file exists in destination OR merge with another file

'pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_MERGE_FILE_TYPE, MergeFile.MERGE_FILE_OTHER)

'Append to existing PDF, insert at the beginning or insert at a page number

'pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_MERGE_TYPE, MergeType.MERGE_TYPE_APPEND)

'if merge with other file, specify file name (and path)

'pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_MERGE_FILE, "invoice-text.pdf")

'if merge type is set to insert at page number

'pNova.SetOverrideOptionLong(NovaOptions.NOVAPDF_OVERRIDE_INSERT_PAGENO, 1)

'if the existing PDF is encrypted

'pNova.SetOverrideOptionEncryptedString(NovaOptions.NOVAPDF_OVERRIDE_PDF_PASSWD, "password")

'Document info

pNova.SetOverrideOptionBool(NovaOptions.NOVAPDF_OVERRIDE_DOC_INFO, 1)

pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_DOC_TITLE, "sdk title")

pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_DOC_SUBJECT, "sdk subject")

pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_DOC_AUTHOR, "sdk author")

pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_DOC_KEY, "sdk key")
pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_DOC_CREATOR, "sdk creator")

' Encrypt PDF

'pNova.SetOverrideOptionBool(NovaOptions.NOVAPDF_OVERRIDE_SECURITY, 1)

'pNova.SetOverrideOptionEncryptedString(NovaOptions.NOVAPDF_OVERRIDE_SECURITY_U, "user")

'pNova.SetOverrideOptionEncryptedString(NovaOptions.NOVAPDF_OVERRIDE_SECURITY_O, "owner")

' Email

pNova.SetOverrideOptionBool(NovaOptions.NOVAPDF_OVERRIDE_SEND_EMAIL, 1)
pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_REC_TO, "to"
)
pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_REC_FROM, "from"
)
pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_REC_CC, "cc"
)
pNova.SetOverrideOptionString(NovaOptions.NOVAPDF_OVERRIDE_REC_BCC, "bcc"
)
pNova.RegisterNovaEvent(NovaEvents.NOVAPDF_EVENT_ACTIONS)

' print a test page
Dim pd As PrintDocument = New PrintDocument
pd.PrinterSettings.PrinterName = PRINTER_NAME
AddHandler pd.PrintPage, AddressOf PrintPageFunction
pd.Print()

Dim bTimeout As Integer
pNova.WaitForNovaEvent(120000, bTimeout)

Catch e As System.Runtime.InteropServices.COMException
    MessageBox.Show(e.Message)
Catch e As Exception
    MessageBox.Show(e.Message)
End Try
End Sub

' and finally the function that actually prints the page
Private Sub PrintPageFunction(ByVal sender As Object, ByVal ev As PrintPageEventArgs)
    Dim str As String = "novaPDF says Hello World from VB.Net"
    Dim font As Font = New Font("Arial", 16)
    Dim brush As Brush = New SolidBrush(Color.Black)
    ev.Graphics.DrawString(str, font, brush, 20.0!, 20.0!)
    ev.HasMorePages = False
End Sub

End Module
1.5.9 Install

1.5.9.1 Installation package bundle

This is a sample on how to compress the msi installations packages in a bundle and generate an setup executable to install novaPDF 11 SDK Developer.

If you haven't already done so, you will need to install the Wix Toolset to build this project.

The project contains two files:

1. Variables.wxi - contains variables for application name, company name, version,...; it's important to change the UpgradeCode to an unique new GUID for your application

2. Bundle.wxs - contains the msi files that need to be included and the execution order; .Net Framework 4 is required by novaPDF 11 SDK Developer and is set as prerequisite

Variables.wxi

```xml
<?xml version="1.0" encoding="utf-8"?>
<Include>
  <!-- change the UpgradeCode to an unique new GUID for your application -->
  <define BundleName="My SDK Application Bundle"/>

  <define AboutUrl='http://www.novapdf.com'/>
  <define SplashImage=" "/>
  <define IconFile=" "/>

  <define MyLicenseFileName="C:\ProgramData\Softland\novaPDF 11\nPdfSdk11_Softland\nPdfSdk11_SoftlandEulaExt.rtf"/>

  <!-- version -->
  <define MajorVersion="11"/>
  <define MinorVersion="0"/>
  <define BuildNumber="1"/>

  <!-- manufacturing -->
  <define Manufacturer="My SDK Company Name"/>

  <!-- product name -->
  <define ProductName="My SDK Application Name"/>

  <!-- full product name -->
  <define FullProductName="$(var.ProductName) $(var.MajorVersion)"/>

  <!-- bundle specific -->
  <define DriverKit86="C:\Users\Public\Documents\novaPDF 11\SDK\Branding\novaPDF11PrinterDriver(x86).msi"/>
  <define DriverKit64="C:\Users\Public\Documents\novaPDF 11\SDK\Branding\novaPDF11PrinterDriver(x64).msi"/>

  <define COMPathx64="C:\Users\Public\Documents\novaPDF 11\SDK\Branding\novaPDF11COM(x64).msi"/>
  <define COMPathx86="C:\Users\Public\Documents\novaPDF 11\SDK\Branding\novaPDF11COM(x86).msi"/>
```
<?define OemKit64="C:\Users\Public\Documents\novaPDF 11\SDK\Brandling\novaPDF11SDK(x64).msi"?>
<?define ToolsKit="C:\Users\Public\Documents\novaPDF 11\SDK\Brandling\novaPDF11Tools.msi"?>
</Include>

Bundle.wxs

<?xml version="1.0" encoding="UTF-8"?>
<Wix xmlns="http://schemas.microsoft.com/wix/2006/wi"
 xmlns:util="http://schemas.microsoft.com/wix/UtilExtension"
 xmlns:bal="http://schemas.microsoft.com/wix/BalExtension"
 xmlns:swid="http://schemas.microsoft.com/wix/TagExtension"
 xmlns:dotNet="http://schemas.microsoft.com/wix/NetFxExtension">

<?include "Variables.wxi"?>

<Bundle Name="$(var.BundleName)"
 Manufacturer="$(var.Manufacturer)"
 UpgradeCode="$(var.UpgradeCode)"
 AboutUrl='$(var.AboutUrl)'
 SplashScreenSourceFile="$(var.SplashImage)"
 Compressed='yes'
 IconSourceFile="$(var.IconFile)" >

<swid:Tag Regid="regid.2008-09.org.wixtoolset" />

<!--change the UpgradeCode to an unique new GUID for your application-->
<RelatedBundle Id="$(var.UpgradeCode)" Action="Upgrade"/>

<!--application license-->
<BootstrapperApplicationRef Id="WixStandardBootstrapperApplication.Rtflicensure" />

<!-- SuppressOptionsUI="yes" />
</BootstrapperApplicationRef>

<Chain>
<!--prereqs-->
 PACKAGEGROUPREF "NetFx40Web" />

</Chain>
Index

- A -
  Addbookmarkdefinition 72
  Addbookmarkdefinition2 73
  Addprofile 81
  Addprofile2 81
  AddWatermarkImage 84
  AddWatermarkImage2 85
  AddWatermarkText 85
  AddWatermarkText2 86
  asp 218
  asp.net 218

- C -
  Choose sample 215
  Components 12
  Copyprofile 96
  Copyprofile2 96
  CSharp converter 239

- D -
  Deletebookmarkdefinition 97
  Deleteprofile 101
  Deleteprofile2 102
  DeleteWatermarkImage 103
  DeleteWatermarkImage2 103
  DeleteWatermarkText 103
  DeleteWatermarkText2 104

- G -
  Getactiveprofile 116
  Getactiveprofile2 117
  Getbookmarkdefinition 117
  Getbookmarkdefinition2 118
  Getbookmarkdefinitioncount 119
  Getfirstprofile 122
  Getfirstprofile2 123
  Getnextprofile 131
  Getnextprofile2 132
  Getoptionlong 134
  Getoptionstring 134
  Getoptionstring2 135
  Getpdffilename 146
  GetWatermarkImage 151
  GetWatermarkImage2 152
  GetWatermarkImageCount 152
  GetWatermarkText 163
  GetWatermarkText2 164
  GetWatermarkTextCount 164

- H -
  hello world 218, 250, 275
  Hello World CSharp 236
  Hello World Delphi 225
  Hellow World network 254
  Hellow World VBNet 286

- I -
  Initialize 164
  Initialize2 165
  Initializeoleusage 165
  InitializeSilent 165
  InitializeSilent2 166
  INovaPdfOptions 64
  Install
    Command line 11
    Network 11
  Integrate 16

- J -
  java 275, 280
  java sample 275, 280

- L -
  LicenseApplication 166
  Licenseoleserver 167
  Licenseshellexecutefile 167

- M -
  Make the release build 16
- MFC Converter 258
- MFC Scribble 262
- Modifybookmarkdefinition 168
- Modifybookmarkdefinition2 169

- N -
- Network auto-install 14

- O -
- ole 280

- P -
- PDF reports 216
- Private/Public Profiles 29

- R -
- Register COM 27
- Register messages 29
- Registereventwindow 170
- RegisterNovaEvent 171
- RegisterNovaEvent2 171
- Registry Keys 33
- Restoredefaultprinter 171

- S -
- Set printer options 29
- Setactiveprofile 180
- Setactiveprofile2 180
- Setdefaultprinter 181
- Setoptionlong 189
- Setoptionstring 189
- Setoptionstring2 190
- SetPrinterOption 199
- System requirements 12

- T -
- Terminal services 11

- U -
- Uninstall 11
- Unregistereventwindow 215
- Use COM 28
- Use Events 30

- V -
- VCL converter 221

- W -
- WaitForNovaEvent 215
- Windows messages 63
- word ole 280
- Word OLE CSharp 241
- Word OLE Delphi 231