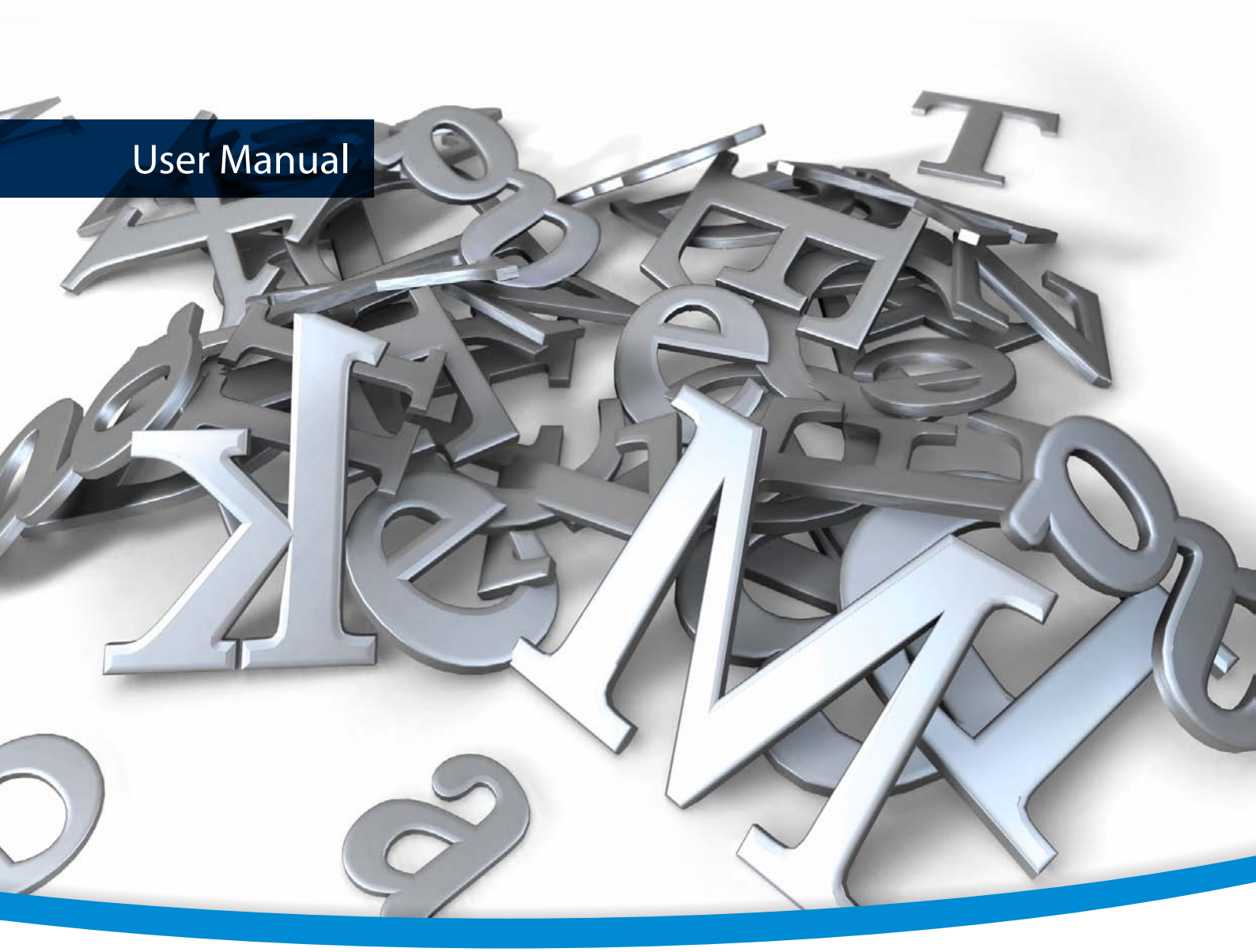
User Manual



3-Heights®
PDF Toolbox API

Version 6.12.10

PDF-TOOLS.COM
Premium PDF Technology

Contents

1 Introduction

1.1 Description

Note: This product will be supported until end 2022 and is superseded by the 4-Heights™ PDF Toolbox.

The 3-Heights® PDF Toolbox API is a component to create, extract, assemble, and modify PDF documents. The interface is based on a comprehensive object model, which reflects the functional range of PDF and PDF/A.

The interface is designed in a way that it can be easily enhanced. It will continuously be adapted to new requirements, such as the evolvement of the PDF/A standard. The component ensures that all PDF/A provisions are enforced, such as file formatting rules, the embedding of fonts and color profiles, and many more.

1.2 Functions

1.2.1 Features

Document assembly

- Copy pages from existing PDFs
- Copy annotations, form fields, links, logical structure, destinations, outlines, layers
- Flatten annotations, form fields, signatures
- Optimize resources
- Crop and rotate pages
- Free composition of content: overlays, underlays, stamps, transformations
- Encryption: user password, owner password, permissions
- Copy and modify document metadata
- Copy and modify page metadata
- Add embedded files and associated files
- Get and set OpenAction destination

Generation

Document Level

- Create pages
- Create form fields
  - General text fields and comb text fields
  - Check boxes
  - Radio buttons
  - List boxes
  - Combo boxes
- Create new outline items and insert them at any position in the tree
- Destinations: Named and direct destinations in the same document

Page Content Level

- Create new PDF content from scratch
- Apply content to existing pages

Colors

- Device colors: RGB, CMYK and grayscale
- ICC color profiles
- Transparency: alpha and blend mode

Paths

- Single and multi-segment lines
- Rectangle, circle, Bezier curves, ellipse, arc, pie
- Filling, stroking, clipping and combinations thereof
- Line width, cap, join, dash array, dash phase and miter limit
- Inside rule: nonzero winding rule, even/odd rule

Text

- Font size, character spacing, word spacing
- Horizontal scaling, leading, rise
- Enables simple text layouting
- Standard PDF fonts, installed fonts
- Font metrics: italic angle, ascent, descent, cap height, character width
- Unicode characters
- Text stroke: line width, line join and dashes
- Fill and stroke text, invisible text
- Use text as clipping path

Images

- Bi-level: CCITT G3, G3 2D and G4, Flate, LZW, Packbits, uncompressed
- 4 bit and 8 bit grayscale: Flate, LZW, Packbits, JPEG and JPEG-6 (8 bit only), uncompressed
- RGB: Flate, JPEG and JPEG-6, LZW, Packbits, uncompressed

Transformations

- Translation
- Scaling
- Skewing (horizontal, vertical)
- Rotation

Page Annotations

- Web link annotations
- File attachment annotations
- Free text annotation
- Sticky note annotation
- Text stamp annotation

- Custom stamp annotation
- Circle annotation
- Square annotation
- Line annotation
- Poly line annotation
- Polygon annotation
- Free drawing annotation

Modification

Page Content

- Selective copying of content elements (without markup)
- Geometric transformation of content elements

Page Annotations

- Web link annotations' target URIs
- Markup annotations' location, creation/modification date, subject, author, content

Form Fields

- Deletion of fields and modification of field values for
  - General text fields and comb text fields
  - Check boxes
  - Radio buttons
  - List boxes
  - Combo boxes

Extraction

Document and Page

- Document information entries: title, author, subject, keywords, creator, producer, creation date, modification date
- Document XMP metadata
- Document encryption settings
- Embedded files
- Page bounding boxes: media box, crop box, bleed box, trim box, art box
- Page XMP metadata
- Outline item tree: Tree structure, item title, expanded/collapsed
- Destinations: Named and direct destinations in the same document

Content

Page's and group's content elements including

- Bounding box
- Affine transformation

As either of the following:

- Group element
- Image element
  - Width and height in pixel
  - Bits per component
  - Color space
- Image mask element
  - Width and height in pixel
  - Paint for filling the mask
- Path element
  - Alignment box
  - Fill parameters including paint and fill rule
  - Stroke parameters including line paint and line style
- Shading element
- Text element
  - Text fragments
    - Bounding box
    - Affine transformation
    - Unicode string
    - Fill parameters including paint and fill rule
    - Stroke parameters including line paint and line style

Annotations

- Annotations: location
- Mark-up annotation: type, location, creation/modification date, subject, author, content
- Custom stamp annotations: appearance
- Link annotations: location, target destination or URI
- Signature fields: name, location, reason, contact info, date, visibility

AcroForm Form Fields

- Form field identifiers, export names and user names, including form field hierarchy
- Form field export and display content of:
  - Push buttons
  - Check boxes
  - Radio buttons
  - General text fields and comb text fields
  - List boxes
  - Combo boxes

1.2.2 Formats

Supported PDF Formats

- PDF 1.x (PDF 1.0, …, PDF 1.7)
- PDF 2.0
- PDF/A-1, PDF/A-2, PDF/A-3

Supported Image Formats

- BMP
- DIB
- JPEG
- JPEG2000
- JBIG2
- PNG
- GIF
- TIFF

Supported Font Formats

- Type1
- TrueType
- OpenType
- OpenType (CFF)

1.2.3 Conformance

Standards:

- ISO 32000-1 (PDF 1.7)
- ISO 32000-2 (PDF 2.0)
- ISO 19005-1 (PDF/A-1)
- ISO 19005-2 (PDF/A-2)
- ISO 19005-3 (PDF/A-3)

1.3 Interfaces

The following interfaces are available:

- C
- Java
- .NET Framework
- .NET Core[1]

1.4 Operating Systems

The 3-Heights® PDF Toolbox API is available for the following operating systems:

- Windows Client 7+ | x86 and x64
- Windows Server 2008, 2008 R2, 2012, 2012 R2, 2016, 2019 | x86 and x64
- Linux:
  - Red Hat, CentOS, Oracle Linux 7+ | x64
  - Fedora 29+ | x64
  - Debian 8+ | x64
  - Other: Linux kernel 2.6+, GCC toolset 4.8+ | x64
- macOS 10.10+ | x64

[1] Limited supported OS versions. Operating Systems

'+' indicates the minimum supported version.

1.5 How to Best Read this Manual

If you are reading this manual for the first time, i.e. would like to evaluate the software, the following steps are suggested.

1. Read the chapter Introduction to verify this product meets your requirements.
2. Identify what interface your programming language uses.
3. Read and follow the instructions in the chapter Installation and Deployment.
4. In the chapter Zip Archive find your programming language. Please note that not every language is covered in this manual.
   For most programming languages there is sample code available. For a start it is generally best to refer to these samples rather than writing code from scratch.
5. (Optional) Read the chapter User's Guide for general information about the API. Read the Interface Reference for specific information about the functions of the API.

2 Installation and Deployment

2.1 Windows

The 3-Heights® PDF Toolbox API comes as a ZIP archive or as a NuGet package.

The installation of the software requires the following steps.

1. You need administrator rights to install this software.
2. Log in to your download account at `http://www.pdf-tools.com`. Select the product "PDF Toolbox API". If you have no active downloads available or cannot log in, please contact `pdfsales@pdf-tools.com` for assistance.
   You will find different versions of the product available. We suggest to download the version, which is selected by default. A different version can be selected using the combo box.
   The product comes as a Zip Archive containing all files, or as a NuGet Package containing all files for development in .NET.
   There is a 32 and a 64-bit version of the product available. While the 32-bit version runs on both, 32 and 64-bit platforms, the 64-bit version runs on 64-bit platforms only. The ZIP archive as well as the NuGet package contain both the 32-bit and the 64-bit version of the product.
3. If you are using the ZIP archive, do the following. Unzip the archive to a local folder, e.g. `C:\Program Files\PDF Tools AG\`.
   This creates the following subdirectories (see also Zip Archive):

Subdirectory	Description
`bin`	Contains the runtime executable binaries.
`doc`	Contains documentation.
`include`	Contains header files to include in your C/C++ project.
`jar`	Contains Java archive files for Java components.
`lib`	Contains the object file library to include in your C/C++ project.
`samples`	Contains sample programs in various programming languages

4. The usage of the NuGet package is described in section NuGet Package.
5. (Optional) Register your license key using the License Management.
6. Identify which interface you are using. Perform the specific installation steps for that interface described in Interface Specific Installation Steps.
7. Ensure the cache directory exists as described in chapter Special Directories.
8. Make sure your platform meets the requirements regarding fonts described in chapter Fonts.

2.2 Linux and macOS

This section describes installation steps required on Linux or macOS.

The Linux and macOS version of the 3-Heights® PDF Toolbox API provides two interfaces:

- Java interface
- Native C interface

Here is an overview of the files that come with the 3-Heights® PDF Toolbox API:

File Description

Name	Description
`bin/x64/libPdfToolboxAPI.so`	This is the shared library that contains the main functionality. The file's extension differs on macOS (`.dylib` instead of `.so`).
`doc/*.*`	Documentation
`include/*.h`	Contains header files to include in your C/C++ project.
`jar/PdfToolboxAPI.jar`	Java API archive.
`samples`	Example code.

2.2.1 Linux

1. Unpack the archive in an installation directory, e.g. `/opt/pdf-tools.com/`
2. Verify that the GNU shared libraries required by the product are available on your system:

   ```
   ldd libPdfToolboxAPI.so
   ```

   In case the above reports any missing libraries you have three options:
   a. Download an archive that is linked to a different version of the GNU shared libraries and verify whether they are available on your system. Use any version whose requirements are met. Note that this option is not available for all platforms.
   b. Use your system's package manager to install the missing libraries. It usually suffices to install the package `libstdc++6`.
   c. Use GNU shared libraries provided by PDF Tools AG:
      1. Go to http://www.pdf-tools.com and navigate to "Support" →"Utilities".
      2. Download the GNU shared libraries for your platform.
      3. Install the libraries manually according your system's documentation. This typically involves copying them to your library directory, e.g. `/usr/lib` or `/usr/lib64`, and running `ldconfig`.
      4. Verify that the GNU shared libraries required by the product are available on your system now.
3. Create a link to the shared library from one of the standard library directories, e.g:

   ```
   ln -s /opt/pdf-tools.com/bin/x64/libPdfToolboxAPI.so /usr/lib
   ```

4. Optionally register your license key using the license manager.
5. Identify which interface you are using. Perform the specific installation steps for that interface described in Interface Specific Installation Steps.
6. Ensure the cache directory exists as described in chapter Special Directories.
7. Make sure your platform meets the requirements regarding fonts described in chapter Fonts.

2.2.2 macOS

The shared library must have the extension `.jnilib` for use with Java. We suggest that you create a file link for this purpose by using the following command:

```
ln libPdfToolboxAPI.dylib libPdfToolboxAPI.jnilib
```

2.3 Zip Archive

The 3-Heights® PDF Toolbox API provides three different interfaces. The installation and deployment of the software depend on the interface you are using. The table below shows the supported interfaces and examples with which programming languages they can be used.

Interface	Programming Languages
`.NET`	The MS software platform .NET can be used with any .NET capable programming language such as: ■ C# ■ VB .NET ■ J# ■ others For a convenient way to use this interface, see NuGet Package.
`Java`	The Java interface is available on all platforms.
`C`	The native C interface is for use with C and C++. This interface is available on all platforms.

2.3.1 Development

The software developer kit (SDK) contains all files that are used for developing the software. The role of each file with respect to the four different interfaces is shown in table Files for Development. The files are split in four categories:

Req. This file is required for this interface.

Opt. This file is optional. See also table File Description to identify which files are required for your application.

Doc. This file is for documentation only.

Empty field An empty field indicates this file is not used at all for this particular interface.

Files for Development

Name	.NET	Java	C
`bin\‹platform›\PdfToolboxAPI.dll`	Req.	Req.	Req.
`bin\*NET.dll`	Req.		
`bin\*NET.xml`	Doc.		
`doc\*.pdf`	Doc.	Doc.	Doc.
`doc\javadoc\*.*`		Doc.	
`include\pdftoolboxapi_c.h`			Req.
`include\*.*`			Opt.
`jar\PdfToolboxAPI.jar`		Req.	

Files for Development

Name	.NET	Java	C
`lib\‹platform›\PdfToolboxAPI.lib`			Req.[2]
`samples\*.*`	Doc.	Doc.	Doc.

The purpose of the most important distributed files of is described in table File Description.

File Description

Name	Description
`bin\‹platform›\PdfToolboxAPI.dll`	This is the DLL that contains the main functionality (required), where `‹platform›` is either `Win32` or `x64` for the 23-bit or the 64-bit library respectively.
`bin\*NET.dll`	The .NET assemblies are required when using the .NET interface. The files `bin\*NET.xml` contain the corresponding XML documentation for MS Visual Studio.
`doc\*.*`	Various documentations.
`include\*.*`	Contains files to include in your C / C++ project.
`lib\‹platform›\PdfToolboxAPI.lib`	On Windows operating systems, the object file library needs to be linked to the C/C++ project.
`jar\PdfToolboxAPI.jar`	The Java API archive.
`samples\*.*`	Contains sample programs in different programming languages.

2.3.2 Deployment

For the deployment of the software only a subset of the files are required. Which files are required (Req.), optional (Opt.) or not used (empty field) for the three different interfaces is shown in the table below.

Files for Deployment

Name	.NET	Java	C
`bin\‹platform›\PdfToolboxAPI.dll`	Req.	Req.	Req.
`bin\*NET.dll`	Req.		
`jar\PdfToolboxAPI.jar`		Req.	

[2] Not required for Linux or macOS.
[3] These files must reside in the same directory as PdfToolboxAPI.dll.

The deployment of an application works as described below:

1. Identify the required files from your developed application (this may also include color profiles).
2. Identify all files that are required by your developed application.
3. Include all these files into an installation routine such as an MSI file or simple batch script.
4. Perform any interface-specific actions (e.g. registering when using the COM interface).

2.4 NuGet Package

Nuget is a package manager that facilitates the integration of libraries for the software development in .NET. The nuget package for the 3-Heights® PDF Toolbox API contains all the libraries needed, managed and native.

Installation Download the package `PdfTools.PdfToolbox.6.12.10.nupkg` from your account on https://www.pdf-tools.com/ to some suitable location.

In Visual Studio click on "Tools" and then "Options". Select "NuGet Package Manager" and add the location of the downloaded package in "Package Sources".

Right-click on a .NET project in Visual Studio and select "Manage NuGet Packages...". Finally, select the package source that was defined above and browse to the desired package.

Development The package `PdfTools.PdfToolbox.6.12.10.nupkg` contains .NET libraries with versions .NET Standard 1.1, .NET Standard 2.0 and .NET Framework 2.0 and native libraries for Windows, macOS and Linux.

The required native libraries are loaded automatically. All project platforms are supported, including "AnyCPU".

In order to use the software, you must first install a license key for the 3-Heights® PDF Toolbox API. To do this you have to download the product kit and use the license manager in it. See also License Management.

> **Note:** This NuGet package is only supported on a subset of the operating systems supported by .NET Core. See also Operating Systems.

2.5 Interface Specific Installation Steps

2.5.1 Java Interface

The 3-Heights® PDF Toolbox API requires Java version 7 or higher.

For compilation and execution When using the Java interface, the Java wrapper `jar\PdfToolboxAPI.jar` needs to be on the CLASSPATH. This can be done by either adding it to the environment variable CLASSPATH, or by specifying it using the switch `-classpath`:

```
javac -classpath ".;C:\Program Files\PDF Tools AG\jar\PdfToolboxAPI.jar" ^
 sampleApplication.java
```

For execution Additionally the library `PdfToolboxAPI.dll` needs be in one of the system's library directories[4] or added to the Java system property `java.library.path`. This can be achieved by either adding it dynamically

[4] On Windows defined by the environment variable PATH and e.g. on Linux defined by LD_LIBRARY_PATH.

at program startup before using the API, or by specifying it using the switch `-Djava.library.path` when starting the Java VM. Choose the correct subdirectory (`x64` or `Win32` on Windows) depending on the platform of the Java VM[5].

```
java -classpath ".;C:\Program Files\PDF Tools AG\PdfToolboxAPI.jar" ^
 "-Djava.library.path=C:\Program Files\PDF Tools AG\bin\x64" sampleApplication
```

Note that on Linux or macOS, the path separator usually is a colon and hence the above changes to something like:

```
... -classpath ".:/path/to/PdfToolboxAPI.jar" ...
```

2.5.2 .NET Interface

The 3-Heights® PDF Toolbox API does not provide a pure .NET solution. Instead, it consists of a native library and .NET assemblies, which call the native library. This has to be accounted for when installing and deploying the tool.

It is recommended to use the NuGet Package. This ensures the correct handling of both the .NET assemblies and the native library.

Alternatively, the files in the Zip Archive can be used directly in a Visual Studio project targeting .NET Framework 2.0 or later. To achieve this, proceed as follows.

The .NET assemblies (`*NET.dll`) are to be added as references to the project; They are needed at compile time. `PdfToolboxAPI.dll` is not a .NET assembly, but a native library. It is not to be added as a reference to the project. Instead, it is loaded during execution of the application.

For the operating system to find and successfully load the native library `PdfToolboxAPI.dll`, it must match the executing application's bitness (32-bit versus 64-bit) and it must reside in either of the following directories:

- In the same directory as the application that uses the library.
- In a subdirectory `win-x86` or Pathwin-x64 for 32-bit or 64-bit applications respectively.
- In a directory that is listed in the PATH environment variable

In Visual Studio, when using the platforms "x86" or "x64", the above can be achieved by adding the 32-bit or 64-bit `PdfToolboxAPI.dll` respectively as an "existing item" to the project, and setting its property "Copy to output directory" to true. When using the "AnyCPU" platform, then you have to make sure by some other means that both the 32-bit and the 64-bit `PdfToolboxAPI.dll` are copied to subdirectories `win-x86` and `win-x46` of the output directory respectively.

Troubleshooting: TypeInitializationException

The most common issue when using the .NET interface is that the correct native DLL `PdfToolboxAPI.dll` is not found at execution time. This normally manifests when the constructor is called for the first time and an exception of type `System.TypeInitializationException` is thrown.

This exception can have two possible causes, distinguishable by the inner exception (property `InnerException`):

`System.DllNotFoundException` Unable to load DLL `PdfToolboxAPI.dll`: The specified module could not be found.

`System.BadImageFormatException` An attempt was made to load a program with an incorrect format.

The following sections describe in more detail, how to resolve the respective issue.

[5] If the wrong data model is used, there is an error message similar to this: "`Can't load IA 32-bit .dll on a AMD 64-bit platform`"

Troubleshooting: DllNotFoundException

This means, that the native DLL `PdfToolboxAPI.dll` could not be found at execution time.

Resolve this by either:

- using the NuGet Package.
- adding `PdfToolboxAPI.dll` as an existing item to your project and set its property "Copy to output directory" to "Copy if newer", or
- adding the directory where `PdfToolboxAPI.dll` resides to the environment variable `%Path%`, or
- manually copying `PdfToolboxAPI.dll` to the output directory of your project.

Troubleshooting: BadImageFormatException

The exception means, that the native DLL `PdfToolboxAPI.dll` has the wrong "bitness" (i.e. platform 32 vs. 64 bit). There are two versions of `PdfToolboxAPI.dll` available in the Zip Archive: one is 32-bit (directory `bin\Win32`) and the other 64-bit (directory `bin\x64`). It is crucial, that the platform of the native DLL matches the platform of the application's process.

(Using the NuGet Package normally ensures that the matching native DLL is loaded at execution time.)

The platform of the application's process is defined by the project's platform configuration for which there are 3 possibilities:

AnyCPU This means, that the application will run as a 32-bit process on 32-bit Windows and as 64-bit process on 64-bit Windows. When using AnyCPU, then a different native DLL has to be used, depending on the Windows platform. This can be ensured either when installing the application by installing the matching native DLL, or at application start-up by determining the application's platform and ensuring the matching native DLL is loaded. The latter can be achieved by placing both the 32 bit and the 64 bit native DLL in subdirectories `win-x86` and `win-x64` of the application's directory respectively.

x86 This means, that the application will always run as 32-bit process, regardless of the platform of the Windows installation. The 32-bit DLL runs on all systems.

x64 This means, that the application will always run as 64-bit process. As a consequence the application will not run on a 32-bit Windows system.

2.5.3 C Interface

- The header file `pdftoolboxapi_c.h` needs to be included in the C/C++ program.
- On Windows operating systems, the library `PdfToolboxAPI.lib` needs to be linked to the project.
- The dynamic link library `PdfToolboxAPI.dll` needs to be in a path of executables (e.g. on the environment variable `%PATH%`).

2.6 Uninstall, Install a New Version

If you have used the ZIP file for the installation: In order to uninstall the product, undo all the steps done during installation, e.g. un-register using `regsvr32.exe /u`, delete all files, etc.

Installing a new version does not require to previously uninstall the old version. The files of the old version can directly be overwritten with the new version. If using the COM interface, the new DLL must be registered, un-registering the old version is not required.

2.7 Fonts

When text is created by the 3-Heights® PDF Toolbox API, all fonts from the Font Directories can be used.

Note that on Windows when a font is installed it is by default installed only for a particular user. It is important to either install fonts for all users, or make sure the 3-Heights® PDF Toolbox API is run under that user and the user profile is loaded.

2.7.1 Font Cache

A cache of all fonts in all Font Directories is created. If fonts are added or removed from the font directories, the cache is updated automatically.

In order to achieve optimal performance, make sure that the cache directory is writable for the 3-Heights® PDF Toolbox API. Otherwise the font cache cannot be updated and the font directories have to be scanned on each program startup.

The font cache is created in the subdirectory `<CacheDirectory>/Installed Fonts` of the Cache Directory.

2.8 Note about the Evaluation License

With the evaluation license the 3-Heights® PDF Toolbox API automatically adds a watermark to the output files.

2.9 Special Directories

2.9.1 Directory for temporary files

This directory for temporary files is used for data specific to one instance of a program. The data is not shared between different invocations and deleted after termination of the program.

The directory is determined as follows. The product checks for the existence of environment variables in the following order and uses the first path found:

Windows

1. The path specified by the `%TMP%` environment variable.
2. The path specified by the `%TEMP%` environment variable.
3. The path specified by the `%USERPROFILE%` environment variable.
4. The Windows directory.

Linux and macOS

1. The path specified by the `$PDFTMPDIR` environment variable.
2. The path specified by the `$TMP` environment variable.
3. The `/tmp` directory.

2.9.2 Cache Directory

The cache directory is used for data that is persisted and shared between different invocations of a program. The actual caches are created in subdirectories. The content of this directory can safely be deleted to clean all caches.

This directory should be writable by the application, otherwise caches cannot be created or updated and performance will degrade significantly.

Windows

- If the user has a profile:
  `%LOCAL_APPDATA%\PDF Tools AG\Caches`
- If the user has no profile:
  `<TempDirectory>\PDF Tools AG\Caches`

Linux and macOS

- If the user has a home directory:
  `~/.pdf-tools/Caches`
- If the user has no home directory:
  `<TempDirectory>/pdf-tools/Caches`

where `<TempDirectory>` refers to the Directory for temporary files.

2.9.3 Font Directories

The location of the font directories depends on the operating system. Font directories are traversed recursively in the order as specified below.

If two fonts with the same name are found, the latter one takes precedence, i.e. user fonts will always take precedence over system fonts.

Windows

1. `%SystemRoot%\Fonts`
2. User fonts listed in the registry key `\HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Fonts`. This includes user specific fonts from `C:\Users\<user>\AppData\Local\Microsoft\Windows\Fonts` and app specific fonts from `C:\Program Files\WindowsApps`
3. directory `Fonts`, which must be a direct sub-directory of where `PdfToolboxAPI.dll` resides.

macOS

1. `/System/Library/Fonts`
2. `/Library/Fonts`

Linux

1. `/usr/share/fonts`
2. `/usr/local/share/fonts`
3. `~/.fonts`
4. `$PDFFONTDIR` or `/usr/lib/X11/fonts/Type1`

3 License Management

The 3-Heights® PDF Toolbox API requires a valid license in order to run correctly. If no license key is set or the license is not valid, then most of the interface elements documented in Interface Reference will fail with an error code and error message indicating the reason.

More information about license management is available in the license key technote.

3.1 License Features

The functionality of the 3-Heights® PDF Toolbox API contains several areas to which the following license features are assigned:

`Assembly` Document assembly and imposition.

`Creation` Creation of documents, pages, and page content.

`Extraction` Extraction of all content and document properties.

`Annotation` Extractpage annotations

`Modification` Modification of page or group content.

`Forms` Creation and modification of form fields.

A license can include an arbitrary set of these features. The presence of any feature in a given license key can be checked in the license manager. The Interface Reference specifies in more detail which functions are included in which license features.

4 User's Guide

4.1 General Concepts

4.1.1 Document model

The document model of the PDF Toolbox API consists of two different types of objects:

Structure objects define the structure of the document, such as `Document`, `Page` or `Content`.

Graphics resources can be used to draw content with a ContentGenerator. Examples are `Image`, `Font` or `ColorSpace`.

All objects in the document model are bound to a specific document. They can only be used in the context of the document for which they were created using one of the `Create..` or `Copy..` method.

The objects of the document model are all stateless. Where a stateful interface is useful, it is provided by an external generator or extractor, which is not considered part of the document model.

4.1.2 Copying instead of modification

The PDF Toolbox API does not allow in-place modification of documents. Instead the content is copied into a new document, while performing the necessary changes.

To copy objects from a source document into a target document, one of the `Copy..` methods is called on the **target** document.

This concept allows processing of very large files without consuming much memory: The content of the input document is only read on demand and any modifications can be directly stored in the output file.

4.1.3 Differentiation between object creation and use

To provide a uniform interface, many operations are divided into two steps:

1. Create (or copy) the object
2. Use the object

This separation allows to provide multiple variants for both steps, without having a "combinatorical explosion" of methods.

Step 1: Create

In the first step, the object is created in the target document or copied from the source document to the target document.

After creating, the object is associated with the document, but not yet used. This means, that copying or creating an object may change the size of the target file, but logically, the PDF is still unchanged.

Examples are the following methods of a `Document`:

- `CreatePage`
- `CreateFont`
- `CopyPage`
- `CopyColorSpace`

- `CopyMetadata`
- `CopyContentElement`

Step 2: Use

The associated object can then be used in the target document.

This second step is often more lightweight that the first step, since all the necessary copying is already done.

Examples are the following methods of a `ContentGenerator`:

- `PaintImage`
- `PaintGroup`
- `AppendContentElement`

or the `Add` method of the `PageList` interface.

4.1.4 Generator Objects

Some objects in a PDF consist of a list or stream of operations that operate on an internal state:

- Content streams
- Text objects
- Path objects

Since all data objects in the PDF Toolbox API are stateless, a (simplified) stateful interface is provided by so called generator interfaces:

- `Content` objects can be modified with a `ContentGenerator`.
- `Path` objects can be modified with a `PathGenerator`.
- `Text` objects can be modified with a `TextGenerator`.

Generator objects must always be closed explicitly, before the associated object can be used.

4.1.5 Garbage collection and closing objects

Every interface object is considered being a resource that needs to be closed after use. Most objects are closed automatically, at the latest when the owning document is closed, in C# and Java possibly earlier by the garbage collector.

In addition to `Document` objects, the "generator" objects `ContentGenerator`, `PathGenerator`, and `TextGenerator` must be closed, lest the generated objects are incomplete.

4.2 Thread safety

The PDF Toolbox API is generally thread-safe with one exception:

A document may only be accessed in one thread concurrently, including all sub-objects.

Note that almost all objects are directly or indirectly associated with a document.

> **Note:** Methods that copy from a source to a target document have to access both documents. The thread safety rules not only apply to the target document, but also to the source document.
> This means that concurrently copying from the same source document is not allowed.

4.2.1 Garbage Collection and Finalizer

Object finalization is thread-safe with one exception:

The finalizer of the `Document` is not thread-safe regarding access to its sub-objects.

Sub-objects do not retain their associated document object. If all references to an open document go out of scope, the document finalizer will eventually be running and the document will be closed.

Explicitly accessing (even closing) any sub-object while the document finalizer is running is **not** safe!

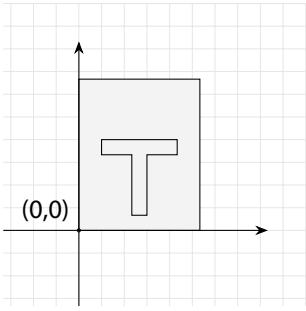
4.3 The PDF Graphics Model

4.3.1 Coordinate System

PDF coordinates are measured from bottom to top, in contrast to many other coordinate systems used in informatics.

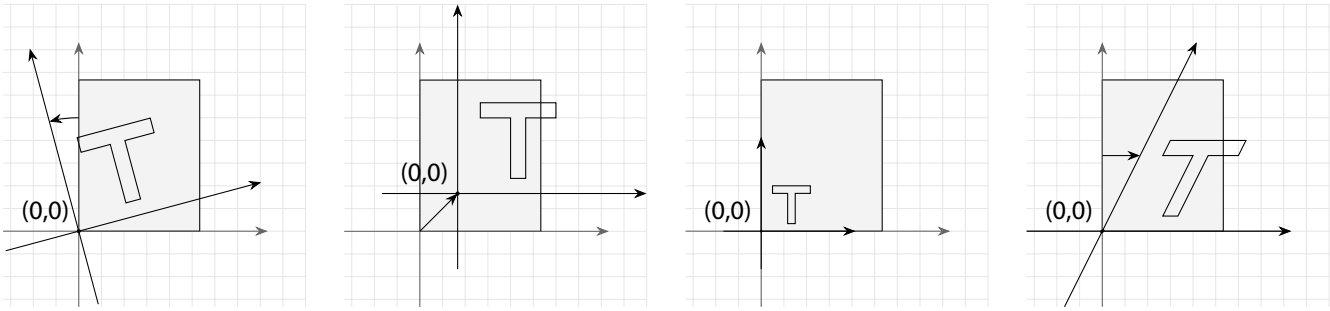
For the sake of simplicity, all coordinates used in the PDF Toolbox API are normalized, such that the point (0,0) denotes the lower left corner of the visible page (crop box).

The internal **Rotate** attribute of a PDF page is not exposed at the API. Instead, all coordinates are assumed to refer to the already rotated page.



4.3.2 Transformations

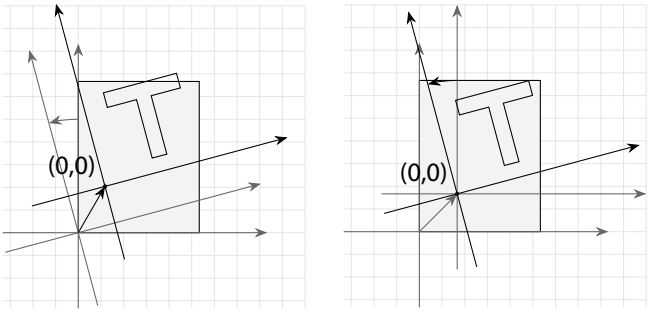
Transformations can be used to rotate, move, scale, or otherwise skew any page content.



Rotate **Move** **Scale** **Skew**

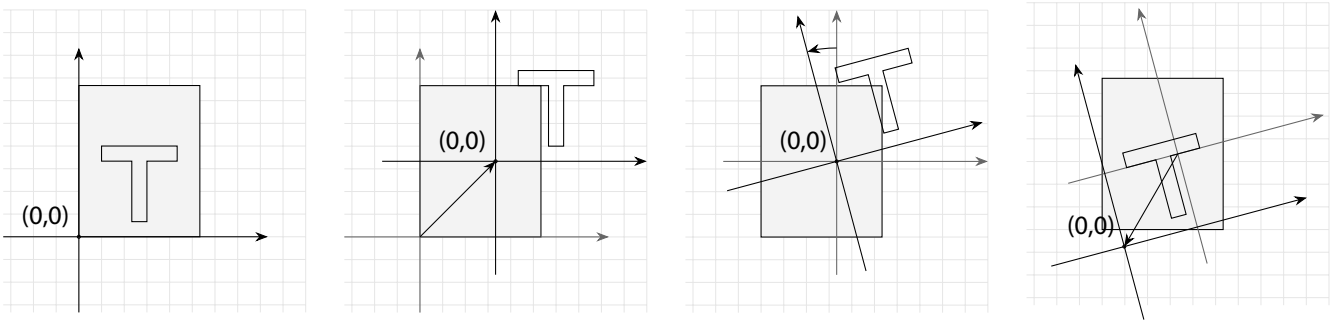
Transformations always affect the coordinate system as a whole. All **following** graphics operations are executed in the transformed coordinate system, including additional transformations.

This means, that the ordering how transformations are applied is important.



Rotate, then Move **Move, then Rotate**

Example: Rotate around a certain point



Start .. **.. Move to center point ..** **.. Rotate ..** **.. Move back to origin**

4.4 Annotations and Form Fields

Page annotations are elements that are not part of a page's content but are applied on top of a page. In contrast to ordinary page content, many annotation types are meant to behave interactively in a PDF viewer.

4.4.1 Form Fields

AcroForm form fields in a PDF consist of data structures that represent variable values, potentially to be modified by a user in a PDF viewing application.

Form fields are structured in a tree topology in which the `Document` acts as the tree root. The immediate child form field nodes thereof can be accessed via the `FormFields` property.

Each child form field node in the tree can itself have more children. We name such a form field a "sub form". A child form field node that has no children is simply a "form field", of which different concrete sub-types are conceived:

- General text field
- Comb text field
- Check box field
- Push button field
- Radio button field
- Combo box field
- List box field

In the PDF Toolbox API, the above types are modeled by interfaces that inherit from a base interface `FormFieldNode` in the following way:

```
FormFieldNode
├─ FormField
│   ├─ CheckBoxField
│   ├─ ChoiceField
│   │   ├─ ComboBoxField
│   │   └─ ListBoxField
│   ├─ PushButtonField
│   ├─ RadioButtonField
│   └─ TextField
│       ├─ CombTextField
│       └─ GeneralTextField
└─ SubForm
```

FormFieldNodes are contained in a FormFieldNodeMap in the form of key-value pairs for which the keys act as form field node **identifiers**. Such an identifier is not allowed to contain any full stops (“.”).

The **fully qualified identifier** of a form field node is defined as the concatenation of all its ancestor SubForm’s identifiers and its own identifier, separated by full stops (“.”), e.g. MySubForm.MyField. The fully qualified identifier of each form field node is unique within a document.

While the form field tree models the form’s data, the visual manifestations of form fields are managed by Widget page Annotations, of which each form field has at least one.

Creating Form Fields

In an output document (a document created with Create), form fields can be created from scratch by means of the following methods:

- CreateCheckBoxField
- CreateComboBoxField
- CreateCombTextField
- CreateGeneralTextField
- CreateListBoxField
- CreateRadioButtonField
- CreateSubForm

A PushButtonField cannot be created.

After creating a ChoiceField, i.e. a ComboBoxField or a ListBoxField, ChoiceItems should be created and added with the choice field’s method AddNewItem.

After creating a RadioButtonField, new RadioButtons should be created and added with the radio button field’s method AddNewButton.

Each created form field must be added either to the document’s FormFields, or to the Children of a SubForm.

It is advisable to set all form field properties prior to creating any Widgets. Specifically, changing form field properties that affect the form field’s visual appearance fails with an UnsupportedOperation error if the form field has widgets.

For each form field, at least one Widget should be created using the radio button’s method AddNewWidget for radio button fields, or the form field’s method AddNewWidget for all other field types.

Finally, each created widget must be added to one of the Annotations of any of the document’s Pages.

A page can either be created from scratch with the CreatePage method or it can be copied with CopyPage from an input document (a document created with Open). In the latter case, the CopyFormFields flag in the CopyOption argument must be **cleared**.

The combination of creating form fields and copying form fields via `CopyPage` (with `CopyOption` argument in which the `CopyFormFields` flag is set) is not supported. Specifically:

- Once `CopyPage` has been called with the `CopyFormFields` flag set, any subsequent call to any of the form field creation methods fails with an `IllegalState` error.
- Once any of the form field creation methods has been called, any subsequent call to `CopyPage` with the `CopyFormFields` flag set fails with an `IllegalState` error.

Filling Form Fields

Filling a form means that the values (field content) of form fields are modified. Depending on the field type this implies the following:

- `TextField`: modify the `Text`.
- `CheckBoxField`: modify the `Checked` state.
- `RadioButtonField`: modify the `ChosenButton`.
- `ComboBoxField`: modify the `ChosenItem` or the `EditableItemName`.
- `ListBoxField`: modify the `ChosenItems`.

In order to use the PDF Toolbox API for filling out the values of form fields in a PDF, the following procedure must be followed:

1. An input document is opened with `Open` and an output document is created with `Create`.
2. Before copying pages, the form fields must be copied from the input document to the output document as follows:
   a. Access the form field node map of the input and the output document via the `FormFields` property.
   b. Copy each form field node found in the input to the output document using the `CopyFormFieldNode` method. Copying `SubForm`s automatically copies their children. (Note that the copied form fields have no `Widgets` yet.)
   c. The value of a copied form field can be modified here or later in Step 3.
   d. Add each copied form field node to the output document's `FormFields` map, preferably using the same key as used in the input document's `FormFields` map.
3. The output document's form field nodes can be accessed, e.g. using the `Lookup` method, to modify form field values.
4. Copy all pages with the `CopyPage` method. Hereby, the `CopyFormFields` flag in the `CopyOption` argument must be **cleared**. In this step, the `Widget`s of input form fields are copied to the output form fields automatically. (As soon as a form field has `Widgets`, its value cannot be modified anymore.)

The combination of filling out form fields and copying form fields via `CopyPage` (with `CopyOption` argument in which the `CopyFormFields` flag is set) is not supported. Specifically:

- Once `CopyPage` has been called with the `CopyFormFields` flag set, any subsequent call to `CopyFormFieldNode` fails with an `IllegalState` error.
- Once `CopyFormFieldNode` has been called, any subsequent call to `CopyPage` with the `CopyFormFields` flag set fails with an `IllegalState` error.

5 Programming Interfaces

Where possible and useful the 3-Heights® PDF Toolbox API uses language specific features. This means that some parts of the API use different syntax for different programming languages.

5.1 .NET Interface

5.1.1 IDisposable Objects

Objects that must be closed explicitly implement the `IDisposable` interface. Instead of calling `Dispose()` directly, it is recommended to use the "`using`" statement:

```
using (Document document = ...)
{
  ...
} // document.Dispose() is called implicitly here
```

See also Garbage collection and closing objects.

5.1.2 Error handling

Errors are reported using exceptions.

The three logic error codes are mapped to the corresponding native exception classes:

`IllegalArgument` maps to `System.ArgumentException`

`IllegalState` maps to `System.InvalidOperationException`

`UnsupportedOperation` maps to `System.NotSupportedException`

The rest of the error codes is modeled using a single exception class `PdfTools.ErrorCodeException` that provides access to the underlying error code and message.

5.1.3 Streams

The native stream interface `System.IO.Stream` is used.

5.1.4 Lists

Lists implement the native list interface `System.Collections.Generic.IList<T>`.

5.1.5 Enumerables

Enumerables implement the native interface `System.Collection.Generic.IEnumerable<T>`.

5.1.6 Maps

Maps implement the native dictionary interface `System.Collections.Generic.IDictionary<K, V>`.

5.2 Java Interface

5.2.1 AutoCloseable Objects

Objects that must be closed explicitly implement the `AutoCloseable` interface. Instead of calling `close()` directly, it is recommended to use the "try-with-resources" statement:

```
try (Document document = ...) {
  ...
} // document.close() is called implicitly here
```

See also Garbage collection and closing objects.

5.2.2 Properties

Properties are modeled with setter and getter methods.

5.2.3 Error handling

Errors are reported using exceptions.

The three logic error codes are mapped to the corresponding native runtime exception classes and are not checked:

`IllegalArgument` maps to `java.lang.IllegalArgumentException`

`IllegalState` maps to `java.lang.IllegalStateException`

`UnsupportedOperation` maps to `java.lang.UnsupportedOperationException`

The rest of the error codes is modeled using a single checked exception class `com.pdf_tools.ErrorCodeException` that provides access to the underlying error code and message.

5.2.4 Streams

The native stream interfaces cannot be used, because they are lacking two important features:

- The PDF file format is based on random access. Native Java streams have only limited support for this.
- The ability to read from an output stream is crucial for processing large files (See Chapter 4.1.2).

Instead we provide a custom stream interface `com.pdf_tools.Stream`, which has a similar interface as `java.io.RandomAccessFile`.

An implementation for files is provided, backed by `java.io.RandomAccessFile`.

5.2.5 Lists

Lists implement the native Java list interface `java.util.List`.

5.2.6 Enumerables

Enumerables (lists that only allow iterating) implement the native Java iterator interface `java.util.Iterable`.

5.2.7 Maps

Maps implement the native Java map interface `java.util.Map`.

5.3 C Interface

5.3.1 Namespaces, classes and methods

In most languages, namespaces and classes are used to model the interfaces.

The exception is C, where this is modeled with function prefixes and functions operating on handles. The prefix of all functions of the 3-Heights® PDF Toolbox API is `Pdf`, for types it is `TPdf` and for enum values `ePdf`.

5.3.2 Library Initialization

The first method called must be `PdfInitialize`. Failing to invoke this function results in undefined behavior. Similarly, the last method must be `PdfUninitialize`.

5.3.3 Objects

Objects in the C interface are represented by object handles. After use, all object handles returned by the 3-Heights® PDF Toolbox API must be closed with `PdfClose`.

5.3.4 Properties

Properties are modeled with setter and getter methods.

5.3.5 Error handling

After a having called a method, an error should be detected as follows:

a. If the method's return type is `BOOL` or a pointer and the return value is `FALSE` or `NULL` respectively, then an error has occurred.
b. If the method's return type is other than `BOOL` or a pointer, then `PdfGetLastError` must be called to detect whether an error has occurred.

More information about the error can be retrieved by using the functions `PdfGetLastError` and `PdfGetLastErrorMessage`.

5.3.6 Strings

All functions involving strings are provided in two different flavors:

- UTF-16 function with suffix `W`, using `WCHAR` as parameter type.
- Multibyte character set function with suffix `A`, using `char` as parameter type. The concrete character set that is used depends on the platform:
  - On Windows, the current ANSI code page (`CP_ACP`) is assumed.
  - On Linux or macOS, the current C encoding (`LC_CTYPE`) is used.

In addition to the effective function names with suffix, there's a macro without suffix for each function pair: It either resolves to the `W` variant (if `_UNICODE` is defined), or to the `A` variant (if `_UNICODE` is **not** defined).

Example: Signature of an API string property setter, where ‹String› stands for the property's name:

```
void PdfSet‹String›A(const char* szString);  // Multibyte encoding
void PdfSet‹String›W(const WCHAR* szString); // UTF-16
#ifdef _UNICODE
#define PdfSet‹String› PdfSet‹String›W
#else
#define PdfSet‹String› PdfSet‹String›A
#endif
```

String return values

Functions that return a string are treated specially in C. Instead of returning the string, those functions take a buffer and size as last parameters and write into that buffer. The return value is the amount of data written to the buffer.

To determine the required buffer size, the function has to be called with NULL as argument.

Calling the function with a buffer size that is too small results in a ePdfErrorIllegalState.

Multibyte character set functions (with suffix A) that return a string can fail to encode the string in the current operating systems' encoding. In case of such a failure, the return value is 0 and no error code is set. In order to prevent such failures, it is recommended to use the UTF-16 (W) functions on Windows or to use operating systems with a Unicode code page.

Example: Signature and usage of an API string property getter (error handling is omitted), where ‹String› stands for the property's name:

```
size_t PdfGet‹String›A(char* pBuffer, size_t nBufferSize);
```

```
size_t nBufferSize = PdfGet‹String›A(NULL, 0);
char* pBuffer = malloc(nBufferSize * sizeof char);
nBufferSize = PdfGet‹String›A(pBuffer, nBufferSize);
```

5.3.7 Streams

Streams are modeled by means of a set of callbacks and a context pointer, grouped in a struct TPdfStreamDescriptor.

An implementation for FILE* is provided in the header file pdfdecl.h. (Search for function PdfCreateFILEStreamDescriptor.)

5.3.8 Lists

Lists in C are implemented like any other interface.

List Interface

Every list type provides a subset of the following properties and methods, where ‹ElementType› stands for the type name of the contained elements:

Count

Property (get): `int Count`

The number of elements of the list.

Get

Method: `‹ElementType› Get(int index)`
Error codes: `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

Get an element of the list.

Append

Method: `Append(‹ElementType› element)`
Error codes: `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

Append an element to the list.

5.3.9 Enumerables

Enumerables (lists that only allow iterating) are implemented like any other interface. For every enumerable type `TPdf‹Name›`, an additional iterator type `TPdf‹Name›Iterator` is defined, where `‹Name›` stands for the enumerable's type name.

Enumerable Interface

Every enumerable type provides the following property:

Iterator

Property (get): `‹Name›Iterator Iterator`
Error code (get): `IllegalState`

Get an iterator for this enumerable.

‹Name›Iterator Interface

Every iterator type provides the following, where `‹ElementType›` stands for the type name of the contained elements:

MoveNext

Method: `bool MoveNext()`
Error code: `IllegalState`

Move the iterator to the next element.

Returns:

- `true` if a current `value` is available.
- `false` if the end has been reached and `value` will be `null`.

`value`

Property (get): `‹ElementType› value`
Error code (get): `IllegalState`

Get the current element or `null` if no elements are left.

5.3.10 Maps

Maps (Dictionaries) in C are implemented like any other interface.

Map Interface

Every map type provides a subset of the following properties and methods, where `‹KeyType›` stands for the type name of the key elements and `‹Value›` for the type name of the value elements:

`Size`

Property (get): `int Size`

The number of entries pairs in the map.

`Set`

Method: `Set(‹KeyType› key, ‹ValueType› value)`
Error codes: `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

Set the value for key.

This operation invalidates all positions previously returned by the methods `GetBegin`, `GetEnd`, `GetNext` and `Get`

`Clear`

Method: `Clear()`
Error codes: `IllegalState`, `UnsupportedOperation`

Remove all entries from the map.

This operation invalidates all positions previously returned by the methods `GetBegin`, `GetEnd`, `GetNext` and `Get`

Get

Method: `int Get(‹KeyType› key)`
Error codes: `NotFound`, `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

Get the position of a key in the map. The position can be used to get the corresponding value with `GetValue`.

GetBegin

Method: `int GetBegin()`
Error codes: `IllegalState`, `UnsupportedOperation`

Get the position of the first entry in the map. The order of the entries arbitrary and not significant.

The position can be used to get the corresponding key and value with `GetValue` and `GetValue` only if the returned value is not equal to the one returned by `GetEnd`.

The next position can be retrieved by `GetNext`.

GetEnd

Method: `int GetEnd()`
Error codes: `IllegalState`, `UnsupportedOperation`

Get the end position of the map.

This position does not correspond to an actual entry and `GetKey`, `GetValue` and `GetNext` must not be called with the returned value.

GetNext

Method: `int GetNext(int it)`
Error codes: `IllegalState`, `UnsupportedOperation`

Get the position of the next entry in the map. The order of the entries arbitrary and not significant.

The position can be used to get the corresponding key and value with `GetValue` and `GetValue` only if the returned value is not equal to the one returned by `GetEnd`.

The next position can be retrieved by `GetNext`.

GetKey

Method: `‹KeyType› GetKey(int it)`
Error codes: `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

Get the key of the entry at a position in the map.

GetValue

Method: `‹ValueType› GetValue(int it)`
Error codes: `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

Get the value of the entry at a position in the map.

SetValue

Method: `SetValue(int it, ‹ValueType› value)`
Error codes: `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

Set the value of the entry at a position in the map.

This operation **does not** invalidate positions previously returned by the methods `GetBegin`, `GetEnd`, `GetNext` and `Get`

Remove

Method: `Remove(int it)`
Error codes: `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

Remove the entry at a position in the map.

This operation **does not** invalidate positions previously returned by the methods `GetBegin`, `GetEnd`, `GetNext` and `Get` except for the position used as a parameter to this method, which can only be used to call `GetNext`.

6 Interface Reference

Note: This chapter describes the C# interface of the 3-Heights® PDF Toolbox API. Other interfaces however work similarly, i.e. they have calls with similar names and the call sequence to be used is the same as with C#.
See chapters Java Interface and C Interface for more information.

6.1 Common Elements

Note: In this section common static methods and properties are listed. They can be called in every interface.

6.1.1 `CheckLicense`

Method: `void CheckLicense()`
Static
Error code: `License`

Check if the product is properly licensed.

This method can be used to perform a license checky without actually opening or creating a document, e.g. when starting a GUI application or a service.

Error Code:

`License` The product is not properly licensed.

6.1.2 `LicenseKey`

Property (set): `String LicenseKey`
Static
Error code (set): `License`

Set the license key.

Note: License keys that require activation can only be installed in the license manager. Setting them at runtime is not supported.

Error Code:

`License` The license key is not valid.

6.1.3 ProductVersion

Property (get): `String ProductVersion`
Static

Get the version of the 3-Heights® PDF Toolbox API in the format "A.C.D.E".

6.2 Annotation Interface

Hierarchy:

```
Annotation
├─ Link
│  ├─ EmbeddedPdfLink
│  ├─ InternalLink
│  └─ WebLink
├─ MarkupAnnotation
│  ├─ DrawingAnnotation
│  │  ├─ LineAnnotation
│  │  ├─ PolyDrawingAnnotation
│  │  │  ├─ FreeDrawingAnnotation
│  │  │  ├─ PolyLineAnnotation
│  │  │  └─ PolygonAnnotation
│  │  └─ ShapeDrawingAnnotation
│  │     ├─ CircleAnnotation
│  │     └─ SquareAnnotation
│  ├─ FileAttachmentAnnotation
│  ├─ FreeTextAnnotation
│  ├─ StampAnnotation
│  │  ├─ CustomStampAnnotation
│  │  └─ TextStampAnnotation
│  ├─ StickyNoteAnnotation
│  ├─ TextInsertAnnotation
│  └─ TextMarkupAnnotation
│     ├─ HighlightAnnotation
│     ├─ SquigglyAnnotation
│     ├─ StrikeThroughAnnotation
│     └─ UnderlineAnnotation
└─ Widget
```

An annotation is a potentially interactive element used to annotate a Page.

6.2.1 Rectangle

Property (get): `Rectangle Rectangle`
Error codes (get): `IllegalState`, `Corrupt`

The location on the page.

Error Codes:

`IllegalState` The `Annotation` object or the owning document has already been closed.

`Corrupt` The PDF is corrupt and the annotation cannot be read.

6.2.2 Hidden

Property (get): `bool Hidden`
Error code (get): `IllegalState`

The annotation's visibility.

If `true` then the annotation is present, but is invisible and not available for user interaction. Depending on `DoPrint`, the annotation will, however, still be visible when printing.

Error Code:

`IllegalState` The `Annotation` object or the owning document has already been closed.

6.2.3 DoPrint

Property (get): `bool DoPrint`
Error code (get): `IllegalState`

The annotation's visibility when printing

If `true` then the annotation is not present in a print output of the document.

Error Code:

`IllegalState` The `Annotation` object or the owning document has already been closed.

6.2.4 NoZoom

Property (get): `bool NoZoom`
Error code (get): `IllegalState`

The annotation's scaling behavior.

If `true` then the annotation's visual appearance does not scale with the zoom factor of a PDF viewer.

Error Code:

`IllegalState` The `Annotation` object or the owning document has already been closed.

6.2.5 NoRotate

Property (get): `bool NoRotate`
Error code (get): `IllegalState`

The annotation's rotation behavior.

If `true` then the annotation's visual appearance does not rotate with the rotation set in a PDF viewer.

Error Code:

`IllegalState` The `Annotation` object or the owning document has already been closed.

6.3 AnnotationList Interface

See Programming Interfaces for more information on how lists are modeled and used in the different interfaces the 3-Heights® PDF Toolbox API provides.

6.4 AnnotationPopup Interface

A pop-up for a `MarkupAnnotation`.

6.4.1 IsOpen

Property (get): `bool IsOpen`
Error code (get): `IllegalState`

The pop-up's visibility state.

Error Code:

`IllegalState` The `AnnotationPopup` object or the owning document has already been closed.

6.4.2 Rectangle

Property (get): `Rectangle Rectangle`
Error code (get): `IllegalState`

The pop-up location.

Error Code:

`IllegalState` The `AnnotationPopup` object or the owning document has already been closed.

6.5 CheckBoxField Interface

This interface represents a check box form field.

Hierarchy:

```
FormFieldNode
 └─ FormField
     └─ CheckBoxField
```

6.5.1 Checked

Property (get, set): `bool Checked`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`, `IllegalState`

Specifies the state of the check box: checked (`true`) or unchecked (`false`).

Error Codes:

`IllegalState` The `CheckBoxField` is `ReadOnly`.

`UnsupportedOperation` The document associated with the `CheckBoxField` is read-only.

6.5.2 CheckedExportName

Property (get): `String CheckedExportName`
Error code (get): `IllegalState`

Specifies the name for the checked state when exporting. (`false`).

Error Code:

`IllegalState` The `CheckBoxField` or the owning document has already been closed.

6.6 `ChoiceField` Interface

Hierarchy:

```
FormFieldNode
 └─ FormField
     └─ ChoiceField
         ├─ ComboBoxField
         └─ ListBoxField
```

6.6.1 `Items`

Property (get): `ChoiceItemList Items`
Error code (get): `IllegalState`

The list of `ChoiceItem`s to choose from.

Adding or removing items or clearing the list is not supported.

Error Code:

`IllegalState` The `ChoiceField` or the owning document has already been closed.

6.6.2 `AddNewItem`

Method: `ChoiceItem AddNewItem(String displayName)`
Error codes: `IllegalState`, `UnsupportedOperation`

Creates a new choice item. The item is automatically added to the choice field's `Items`.

Parameter:

`displayName` `[String]` This item's display name.

Returns:

The newly created choice item.

6.7 ChoiceItem Interface

This interface represents an item in a ChoiceField, i.e. in a ListBoxField or in a ComboBoxField.

6.7.1 DisplayName

Property (get, set): String DisplayName
Error code (get): IllegalState
Error codes (set): IllegalState, UnsupportedOperation

Specifies the item's name displayed in the user interface.

Error Codes:

IllegalState

- (set, get) The ChoiceItem or the owning document has already been closed.
- (set) The ChoiceItem has Widgets.

UnsupportedOperation The document is read-only.

6.7.2 ExportName

Property (get, set): String ExportName
Error code (get): IllegalState
Error codes (set): IllegalState, UnsupportedOperation

Specifies the item's name when exporting.

Error Codes:

IllegalState The ChoiceItem or the owning document has already been closed.

UnsupportedOperation The document is read-only.

6.8 ChoiceItemList Interface

See Programming Interfaces for more information on how lists are modeled and used in the different interfaces the 3-Heights® PDF Toolbox API provides.

6.9 CircleAnnotation Interface

A circle or ellipse drawing annotation.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ DrawingAnnotation
         └─ ShapeDrawingAnnotation
             └─ CircleAnnotation
```

6.10 ColorSpace Interface

6.10.1 Components

Property (get): `int Components`
Error code (get): `IllegalState`

The number of components in the color space.

Error Code:

`IllegalState` The `ColorSpace` object or the owning document has already been closed.

6.10.2 Name

Property (get): `String Name`
Error code (get): `IllegalState`

The name of the color space. E.g. `"DeviceRGB"`, `"DeviceCMYK"`, `"Indexed"`, `"Separation"`, `"DeviceN"`, `"Pattern"`, etc.

Error Code:

`IllegalState` The `ColorSpace` object or the owning document has already been closed.

6.10.3 Type

Property (get, set): `ColorSpaceType Type`

The type of the color space.

6.11 ComboBoxField Interface

This interface represents a drop-down combo box with items to choose from.

Hierarchy:

```
FormFieldNode
 └─ FormField
     └─ ChoiceField
         └─ ComboBoxField
```

6.11.1 CanEdit

Property (get, set):	`bool CanEdit`
Error code (get):	`IllegalState`
Error codes (set):	`IllegalState`, `UnsupportedOperation`

When `true`, then the combo box allows the user to enter an arbitrary string as chosen item.

Error Codes:

`IllegalState` The combo box field or the owning document has already been closed.

`UnsupportedOperation` The document associated with the combo box field is read-only.

6.11.2 ChosenItem

Property (get, set):	`ChoiceItem ChosenItem`
Error code (get):	`IllegalState`
Error codes (set):	`IllegalState`, `UnsupportedOperation`, `IllegalArgument`

The selected combo box `ChoiceItem`.

If `null` then the editable item is selected and its value is `EditableItemName`.

Setting this property automatically sets `EditableItemName` to `null`.

Error Codes:

`IllegalState`

- (get, set) The combo box field or the owning document has already been closed.
- (set) The combo box field is `ReadOnly`.
- (set) The combo box field has `Widgets`.

`UnsupportedOperation` The document associated with the combo box field is read-only.

`IllegalArgument`

- The given `ChoiceItem` does not belong to this combo box field.
- The given `ChoiceItem` is `null`.

6.11.3 EditableItemName

Property (get, set):	`String EditableItemName`
Error code (get):	`IllegalState`
Error codes (set):	`IllegalState`, `UnsupportedOperation`

This property represents the freely editable string when `CanEdit` is `true`.

This property is `null` if any of the `Items` is selected, i.e. if `ChosenItem` is not `null`.

Setting this property automatically sets `ChosenItem` to `null`.

Error Codes:

`IllegalState`

- (get, set) The combo box field or the owning document has already been closed.
- (set) The combo box field is `ReadOnly`.
- (set) The `CanEdit` is `false`.
- (set) The combo box field has `Widgets`.

`UnsupportedOperation` The document associated with the combo box field is read-only.

6.12 CombTextField Interface

In contrast to `GeneralTextFields`, a `CombTextField` displays the `Text` as follows: The `Widget`'s `Rectangle` is partitioned into `MaxLength` cells ("combs") of equal width, and the `Text` is displayed by displaying one character per cell.

In a `CombTextField`, `MaxLength` never is undefined.

Hierarchy:

```
FormFieldNode
 └─ FormField
     └─ TextField
         └─ CombTextField
```

6.13 Content Interface

Represents the content of a page or a group.

The interface has no methods on its own, but can be modified with a `ContentGenerator` or extracted with a `ContentExtractor`.

6.14 ContentElement Interface

> **Note:** While content elements incorporate all graphical aspects of page content, they do not include structural information, i.e. "tags". As a consequence, copying content elements from an input page to an output page infers loss of such information. Therefore, documents with a PDF/A conformance level "a" (PDF/A-1a, PDF/A-2a, and PDF/A-3a) cannot be created using `ContentElement`s.

Hierarchy:

```
ContentElement
  ├─ GroupElement
  ├─ ImageElement
  ├─ ImageMaskElement
  ├─ PathElement
  ├─ ShadingElement
  └─ TextElement
```

6.14.1 BoundingBox

Property (get): `Rectangle BoundingBox`
Error code (get): `IllegalState`

This is the rectangle in transformed coordinates that encompasses all the parts of the content element. Use the `Transformation` to compute the untransformed (actual) coordinates with respect to the parent page or group content.

Error Code:

`IllegalState` The `ColorSpace` object or the owning document has already been closed.

6.14.2 Transform

Property (get): `Transformation Transform`
Error code (get): `IllegalState`

This is the coordinate transformation applicable to this content element. Use this transform to compute the actual location of the content element's bounding box on the containing page or group.

Error Code:

`IllegalState` The `ColorSpace` object or the owning document has already been closed.

6.15 ContentExtractor Interface

This interface is an enumerable that can be used to iterate through ContentElements. See Programming Interfaces for more information on how enumerables are modeled and used in the different interfaces the 3-Heights® PDF Toolbox API provides.

6.15.1 ContentExtractor Constructor

Method: ContentExtractor(Content content)
Error codes: IO, Corrupt, IllegalArgument, UnsupportedOperation
License features: Extraction, Modification

Create a new content extractor for extracting a page's or group's ContentElements.

Parameter:

content [Content] The content object of a page or group.

Error Codes:

IO Error occurred reading from the document.

Corrupt The document is corrupt.

IllegalArgument

- The content argument is null.
- The content argument has already been closed.
- The Document object associated with the content argument has already been closed.
- The Page/Group object associated with the content argument has already been closed.

UnsupportedOperation The content argument stems from a group object that has been created with CreateGroup or CopyPageAsGroup.

6.15.2 Ungrouping

Property (get, set): UngroupingSet Ungrouping
Default: None
Error code: IllegalState

Use this property to configure the extractor's behavior regarding the selection of groups to be un-grouped.

Groups in the content can either be extracted as GroupElements, or their content can be un-grouped, in which case groups' content elements are extracted as if not belonging to a group.

6.16 ContentGenerator Interface

6.16.1 ContentGenerator Constructor

Method: `ContentGenerator(Content content, bool prepend)`
Error codes: `IO`, `Corrupt`, `IllegalArgument`
License features: `Assembly`, `Creation`, `Modification`

Create a new content generator for appending or prepending to the content of a page or group.

Parameters:

`content` [`Content`] The content object of a page or group.

`prepend` [`bool`]

- **`true`** for prepending to the content (apply content to background of page).
- **`false`** for appending (apply content to foreground of page).

Error Codes:

`IO` Error occurred reading from the document.

`Corrupt` The document is corrupt.

`IllegalArgument`

- The `content` argument is `null`.
- The `content` argument has already been closed.
- The `Document` object associated with the `content` argument has already been closed.
- The `Page`/`Group` object associated with the `content` argument has already been closed.

6.16.2 AppendContentElement

Method: `void AppendContentElement(ContentElement contentElement)`
Error codes: `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

Paint a `ContentElement`, i.e. either of the following: `GroupElement`, `ImageElement`, `ImageMaskElement`, `PathElement`, `ShadingElement`, or `TextElement`.

Parameter:

`contentElement` [`ContentElement`] The `ContentElement` to be painted.

Error Codes:

`IllegalState`

- The `contentElement` has already been closed.
- The associated `Content` object has already been closed.

`IllegalArgument`

- The `contentElement` is associated with a different document.
- The `contentElement` is `null`.

`UnsupportedOperation` This content generator's content stems from a `Group` object that has been created with `CreateGroup` or `CopyPageAsGroup`.

6.16.3 Close

Method: `void Close()`
Error code: `IllegalState`

Close the Generator.

The generated object is not finalized until this method is called.

All native data structures that belong to the generator are freed.

After closing the generator, any call to a method of the generator will result in an `IllegalState` error.

Error Code:

`IllegalState` The generator has already been closed.

6.16.4 PaintImage

Method: `void PaintImage(Image image, Rectangle targetRect)`
Error codes: `IllegalState`, `IllegalArgument`

Paint an image.

See `Image`.

Parameters:

`image` `[Image]` The image to be painted.

`targetRect` `[Rectangle]` The target rectangle in the current coordinate system. If `targetRect` is `null`, the unit rectangle `[0, 0, 1, 1]` is used.

Error Codes:

IllegalState

- The associated Document object has already been closed.
- The associated Page/Group object has already been closed.
- The associated Content object has already been closed.
- The ContentGenerator object has already been closed.

IllegalArgument

- The image argument is null.
- The image argument has already been closed.
- The image argument is associated with a different Document object.

6.16.5 PaintImageMask

Method: void PaintImageMask(ImageMask imageMask, Rectangle targetRect, Paint fill)
Error codes: IllegalState, IllegalArgument

Paint an image (stencil) mask.

An image mask is a monochrome image, in which each sample is specified by a single bit. However, instead of being painted in opaque black and white, the image mask is treated as a stencil mask that is partly opaque and partly transparent. Sample values in the image do not represent black and white pixels; rather, they designate places on the content that should either be marked with the fill paint or masked out (not marked at all). Areas that are masked out retain their former contents.

The effect is like applying paint in the current color through a cut-out stencil, which allows the paint to reach the page in some places and masks it out in others.

Parameters:

imageMask [ImageMask] The image (stencil) mask.

targetRect [Rectangle] The target rectangle in the current coordinate system. If targetRect is null, the unit rectangle [0, 0, 1, 1] is used.

fill [Paint] The paint for filling marked pixels.

Error Codes:

IllegalState

- The associated Document object has already been closed.
- The associated Page/Group object has already been closed.
- The associated Content object has already been closed.
- The ContentGenerator object has already been closed.

IllegalArgument

- The imageMask argument is null.
- The imageMask argument has already been closed.

- The `imageMask` argument is associated with a different `Document` object.
- The `fill` argument is `null`.
- The `fill` argument has already been closed.
- The `fill` argument is associated with a different `Document` object.

6.16.6 PaintPath

Method: `void PaintPath(Path path, Paint fill, StrokeParams stroke, bool intersectClipPath)`
Error codes: `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

Paint a path.

The path is first filled and then stroked, then the clip path is intersected.

> **Note:** The blend modes for filling and stroking must be the same.

Parameters:

`path` `[Path]` The path to be painted.

`fill` `[Paint]` The fill paint or `null` if the path should not be filled.

`stroke` `[StrokeParams]` The stroke properties or `null` if the path should not be stroked.

`intersectClipPath` `[bool]` If `true`, the current clip path is intersected.

Error Codes:

`IllegalState`

- The associated `Document` object has already been closed.
- The associated `Page`/`Group` object has already been closed.
- The associated `Content` object has already been closed.
- The `ContentGenerator` object has already been closed.

`IllegalArgument`

- The `path` argument is `null`.
- The `path` argument has already been closed.
- The `fill` argument has already been closed.
- The `fill` argument is associated with a different `Document` object.
- The `Paint` object associated with the `stroke` argument is `null`.
- The `Paint` object associated with the `stroke` argument has already been closed.
- The `Paint` object associated with the `stroke` argument is associated with a different `Document` object.

`UnsupportedOperation` The `Paint` objects associated with the `fill` and `stroke` arguments use different blend modes.

6.16.7 PaintText

Method: `void PaintText(Text text)`
Error codes: `IllegalState`, `IllegalArgument`, `Generic`

Paint text.

Parameter:

`text` `[Text]` The text to be painted.

Error Codes:

`IllegalState`

- The associated `Document` object has already been closed.
- The associated `Page`/`Group` object has already been closed.
- The associated `Content` object has already been closed.
- The `ContentGenerator` object has already been closed.

`IllegalArgument`

- The `text` argument is `null`.
- The `text` argument has already been closed.
- The `text` argument is associated with a different `Document` object.

`Generic` For some of the requested characters to paint the font's encoding is not defined or no glyph exists in the font.

6.16.8 PaintGroup

Method: `void PaintGroup(Group group, Rectangle targetRect, TransparencyParams transparency)`
Error codes: `IllegalState`, `IllegalArgument`, `Conformance`

Paint a group.

Parameters:

`group` `[Group]` The group to be painted.

`targetRect` `[Rectangle]` The target rectangle in the current coordinate system. If `targetRect` is `null`, a default rectangle `[0, 0, groupWidth, groupHeight]` is used.

`transparency` `[TransparencyParams]` The transparency parameters to be used when painting the group. if `transparency` is `null`, then the group is painted opaquely.

Error Codes:

`IllegalState`

- The associated `Document` object has already been closed.
- The associated `Page`/`Group` object has already been closed.
- The associated `Content` object has already been closed.
- The `ContentGenerator` object has already been closed.

`IllegalArgument`

- The `group` argument is `null`.
- The `group` argument has already been closed.
- The `group` argument is associated with a different `Document` object.
- The `group` argument contains no content or the content generator has not been closed yet.
- The `group` argument contains interactive elements (see `CopyPageAsGroup`) and has already been painted.
- The `group` argument contains interactive elements (see `CopyPageAsGroup`) and the content of the content generator belongs to an annotation.

`Conformance` The `transparency` argument is not `null` and has properties `ConstAlpha` other than `1.0` or `BlendMode` other than `Normal`, and the explicitly specified conformance does not support transparency (PDF/A-1, PDF 1.0 - 1.3).

6.16.9 Restore

Method: `void Restore()`
Error code: `IllegalState`

Restore the graphics state.

The most recently saved state is restored and removed from the graphics state stack.

The following properties are affected:

- The current transform matrix.
- The current clip path.

Error Code:

`IllegalState`

- The associated `Document` object has already been closed.
- The associated `Page`/`Group` object has already been closed.
- The associated `Content` object has already been closed.
- The `ContentGenerator` object has already been closed.

6.16.10 Save

Method: `void Save()`
Error code: `IllegalState`

Save the current graphics state.

The graphics state is stored on the graphics state stack.

The following properties are affected:

- The current transform matrix.
- The current clip path.

Error Code:

`IllegalState`

- The associated `Document` object has already been closed.
- The associated `Page`/`Group` object has already been closed.
- The associated `Content` object has already been closed.
- The `ContentGenerator` object has already been closed.

6.16.11 Transform

Method: `void Transform(Transformation transformation)`
Error codes: `IllegalState`, `IllegalArgument`

Modify the current transformation matrix by concatenating the specified matrix.

Parameter:

`transformation` [`Transformation`] The transformation that is applied to the current transformation.

Error Codes:

`IllegalState`

- The associated `Document` object has already been closed.
- The associated `Page`/`Group` object has already been closed.
- The associated `Content` object has already been closed.
- The `ContentGenerator` object has already been closed.

`IllegalArgument`

- The `transformation` argument is non-invertible.
- The `transformation` argument is `null`.
- The `transformation` argument has already been closed.

6.17 CustomStampAnnotation Interface

A stamp annotation with custom content.

Hierarchy:

```
Annotation
└─ MarkupAnnotation
    └─ StampAnnotation
        └─ CustomStampAnnotation
```

6.17.1 Appearance

Property (get, set): `Group Appearance`
Error code: `IllegalState`

The custom stamp's visual appearance.

Error Code:

`IllegalState` The `CustomStampAnnotation` object or the owning document has already been closed.

6.18 Destination Interface

A destination is a location in the document that can be used as a jump target, e.g. for outline items (bookmarks) or link annotations.

Hierarchy:

```
Destination
├─ DirectDestination
│   ├─ FitHeightDestination
│   ├─ FitPageDestination
│   ├─ FitRectangleDestination
│   ├─ FitWidthDestination
│   └─ LocationZoomDestination
└─ NamedDestination
```

6.18.1 Target

Property (get): `DirectDestination Target`
Error code (get): `IllegalState`

The target destination.

For a `DirectDestination`, this is just the destination itself.

Error Code:

`IllegalState`

- The destination has already been closed.
- The associated document has already been closed.

6.19 `DirectDestination` Interface

A destination that directly points to a specific location in the document.

> **Note:** Most PDF viewers support different viewing modes like "fit page" or "fit width". Most destination types will change the current viewing mode in those viewers.
> Changing the viewing mode is usually not very well received by users and thus a `LocationZoomDestination` should be preferred in most cases.

Hierarchy:

```
Destination
└─ DirectDestination
    ├─ FitHeightDestination
    ├─ FitPageDestination
    ├─ FitRectangleDestination
    ├─ FitWidthDestination
    └─ LocationZoomDestination
```

6.19.1 Page

Property (get): `Page` `Page`
Error codes (get): `IllegalState`, `Corrupt`

The page in the document that this destination is pointing to.

Error Codes:

`IllegalState`

- The destination has already been closed.
- The associated document has already been closed.

`Corrupt`

- The page could not be found or does not exist in the document.

6.20 Document Interface

6.20.1 AssociatedFiles

Property (get): `FileReferenceList AssociatedFiles`
Error code (get): `IllegalState`
License features: `Extraction`, `Assembly`

This is the list of embedded files that are associated with the document.

If the document is writable, then it is possible to append new `FileReference`s to the list. File references appended to this list are automatically also appended to the `EmbeddedFiles`. Every `FileReference` object can occur at most once in this list.

Appending to this list results in a `Conformance` error if the document's `Conformance` is neither PDF/A-3 nor PDF 2.0. Furthermore, reading from this list is not supported and results in an `UnsupportedOperation` error.

Error Code:

`IllegalState` The document has already been closed.

6.20.2 Close

Method: `void Close()`
Error codes: `IO`, `Corrupt`, `IllegalState`

Close the PDF document.

If the document instance is writable, all remaining data is written to the stream and the document is finalized. All native data structures that belong to the document are freed.

After closing the document, any call to a method of the document or its subobjects (e.g. `Page` objects) will result in an `IllegalState` error.

Error Codes:

`IO` Error writing to the stream.

`Corrupt` The file to be written is corrupt (i.e. contains no pages).

`IllegalState` The document has already been closed.

6.20.3 Conformance

Property (get): `Conformance Conformance`
Error code (get): `IllegalState`

Get the claimed conformance of the document.

This property only returns the **claimed** conformance level, the document is not validated.

Error Code:

`IllegalState` The document has already been closed.

6.20.4 CopyAnnotation

Method: `Annotation CopyAnnotation(Annotation annotation)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`, `Conformance`
License features: `Forms`, `Annotation`

Copy an annotation from a different document.

The newly created annotation is associated with the document, but does not belong to any page yet until added to a `Page`'s `Annotations`.

> **Thread safety:** This method needs **exclusive** access to the source document.

> **License Feature Note:** Copying from multiple input documents is only possible if the license feature `Assembly` is active.

Parameter:

`annotation` `[Annotation]` An annotation of a different document.

Returns:

The copied annotation, associated with the current document.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation`

- The document is read-only.
- The `annotation` argument is not associated with an input document.

`IllegalArgument`

- The `annotation` argument is `null`.
- The `annotation` argument has already been closed.
- The `annotation` argument is a `Widget`.
- The `annotation` argument is not associated with any document.

`Conformance`

- The conformance level of the source document is not compatible with the conformance level of the target document.

6.20.5 CopyColorSpace

Method: `ColorSpace CopyColorSpace(ColorSpace colorSpace)`
Error codes: `IO`, `Corrupt`, `Conformance`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`

Create a new color space object from a color space of a different document.

Thread safety: This method needs **exclusive** access to the source document.

License Feature Note: Copying from multiple input documents is only possible if the license feature `Assembly` is active.

Parameter:

`colorSpace` `[ColorSpace]` A color space of a different document.

Returns:

The copied color space, associated with the current document.

Error Codes:

`IO` Error reading from the source document or writing to the target document.

`Corrupt` The source document is corrupt.

`Conformance`

- The conformance level of the source document is not compatible with the conformance level of the target document.

`IllegalState` The document has already been closed.

`UnsupportedOperation`

- The document is read-only.

- The `colorSpace` argument is not associated with an input document.

`IllegalArgument`

- The `colorSpace` argument is `null`.
- The `colorSpace` argument has already been closed.

6.20.6 `CopyFileReference`

Method: `FileReference CopyFileReference(FileReference fileReference)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`, `Conformance`

Thread safety: This method needs **exclusive** access to the source document.

License Feature Note: Copying from multiple input documents is only possible if the license feature `Assembly` is active.

Parameter:

`fileReference` `[FileReference]` A file reference object of a different document.

Returns:

The copied file reference, associated with the current document.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation`

- The document is read-only.
- The `fileReference` argument is not associated with an input document.

`IllegalArgument`

- The `fileReference` is `null`.
- The `fileReference` object has already been closed.

`Conformance`

- The conformance level of the source document is not compatible with the conformance level of the target document.

6.20.7 CopyContentElement

Method: `ContentElement CopyContentElement(ContentElement contentElement)`
Error codes: `IO`, `Conformance`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Modification`

Copy a content element from a different document.

The newly created content element is associated with the document but is not yet used in any content of the document's pages or groups. Use a `ContentGenerator`'s `AppendContentElement` method to use the content element.

Thread safety: This method needs **exclusive** access to the source document.

License Feature Note: Copying from multiple input documents is only possible if the license feature `Assembly` is active.

Parameter:

`contentElement` `[ContentElement]` A content element associated with a different document.

Returns:

A content element that is a copy of the passed content element, but associated with the current document. The actual type of the return value is either `GroupElement`, `ImageElement`, `ImageMaskElement`, `PathElement`, `ShadingElement`, or `TextElement`.

Error Codes:

`IO` Error reading from the source document or writing to the target document.

`Conformance`

- The conformance level of the source document is not compatible with the conformance level of the target document.

`IllegalState` The document has already been closed.

`UnsupportedOperation` The `contentElement` is not associated with an input document.

`IllegalArgument`

- The `contentElement` argument is `null`.
- The `contentElement` object has already been closed.

6.20.8 CopyFormFieldNode

Method: `FormFieldNode CopyFormFieldNode(FormFieldNode formFieldNode)`
Error codes: `IO`, `Conformance`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Forms`

Copy a form field node from a different document. The newly created form field node is associated with the document but is not yet added to the `FormFields` or to any `SubForm`.

If the form field node is a `SubForm` then all child nodes are copied as well.

Thread safety: This method needs **exclusive** access to the source document.

License Feature Note: Copying from multiple input documents is only possible if the license feature `Assembly` is active.

Parameter:

`formFieldNode` `[FormFieldNode]` A form field node associated with a different document.

Returns:

A form field node that is a copy of the passed form field node, but associated with the current document.

Error Codes:

`IO` Error reading from the source document or writing to the target document

`Conformance`

- The conformance level of the source document is not compatible with the conformance level of the target document.

`IllegalState`

- The document has already been closed.
- The document contains form fields that have been implicitly copied by `CopyPage` with activated `CopyOption` `CopyFormFields`.

`UnsupportedOperation`

- The document is read-only.
- The `formFieldNode` is not associated with an input document.

`IllegalArgument`

- The `formFieldNode` argument is `null`.
- The `formFieldNode` has already been closed.
- The `formFieldNode` argument has already been copied to this document.

6.20.9 CopyGroupElementWithoutContent

Method: `GroupElement CopyGroupElementWithoutContent(GroupElement groupElement)`
Error codes: `IO`, `Conformance`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Modification`

Create a new group element, taking a given group element of a different document as a template. The newly created group has the same properties, such as geometric `Transform`, clipping, and soft mask, but it's content is empty.

This method can be used when copying content recursively.

Thread safety: This method needs **exclusive** access to the source document.

License Feature Note: Copying from multiple input documents is only possible if the license feature `Assembly` is active.

Parameter:

`groupElement` `[GroupElement]` a group element object associated with a different document.

Returns:

A group element object with the same properties as `groupElement`, but without content and associated with the current document.

Error Codes:

`IO` Error reading from the source document or writing to the target document.

`Conformance`

- The conformance level of the source document is not compatible with the conformance level of the target document.

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `groupElement` argument is `null`.
- The `groupElement` object has already been closed.

6.20.10 CopyMetadata

Method: `Metadata CopyMetadata(Metadata metadata)`
Error codes: `IO`, `Corrupt`, `Conformance`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`

Copy a metadata object from a different document.

The newly created metadata object is associated with the document but not automatically used as the document metadata. Set the property `Metadata` to achieve that.

> **Thread safety:** This method needs **exclusive** access to the source document.

> **License Feature Note:** Copying from multiple input documents is only possible if the license feature `Assembly` is active.

Parameter:

`metadata` `[Metadata]` A metadata object associated with a different document.

Returns:

A metadata object with the same content as `metadata`, but associated with the current document.

Error Codes:

`IO` Error reading from the source document or writing to the target document.

`Corrupt` The source document is corrupt.

`Conformance`

- The conformance level of the source document is not compatible with the conformance level of the target document.

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `metadata` argument is `null`.
- The `metadata` object has already been closed.

6.20.11 CopyOutlineItem

Method: `Page CopyOutlineItem(OutlineItem outlineItem, CopyOption options)`
Error codes: `Conformance`, `IllegalState`, `IllegalArgument`, `UnsupportedOperation`
License features: `Modification`, `Assembly`

Create an new outline item from an outline item of a different document.

This method copies the outline item including all descendants.

> **Thread safety:** This method needs **exclusive** access to the source document.

> **License Feature Note:** Copying from multiple input documents is only possible if the license feature `Assembly` is active.

Parameters:

`outlineItem` `[OutlineItem]` An outline item of a different document.

`options` `[CopyOption]` The options used to copy the item.

The only relevant flags for this method are `CopyLogicalStructure` and `CopyNamedDestinations`.

Returns:

The copied outline item, associated with the current document.

Error Codes:

`Conformance`

- The conformance level of the source document is not compatible with the conformance level of the target document.

`IllegalState`

- The document has already been closed.
- The document contains implicitly copied outline items.

`IllegalArgument` The `formFieldNode` argument is `null`.

`UnsupportedOperation`

- The document is read-only.
- The `formFieldNode` argument is not associated with an input document.

6.20.12 CopyPage

Method: `Page CopyPage(Page page, CopyOption options)`
Error codes: `IO`, `Corrupt`, `Conformance`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License features: `Annotation`, `Assembly`, `Forms`, `Modification`

Create a page object from a page of a different document.

Thread safety: This method needs **exclusive** access to the source document.

License Feature Note: Copying from multiple input documents is only possible if the license feature `Assembly` is active.

Parameters:

`page` `[Page]` A page of a different document.

`options` `[CopyOption]` The copy options.

Returns:

The copied page, associated with the current document.

Error Codes:

`IO` Error reading from the source document or writing to the target document.

`Corrupt` The source document is corrupt.

`Conformance`

- The conformance level of the source document is not compatible with the conformance level of the target document.
- The explicitly requested conformance level is PDF/A Level A (`PdfA1A`, `PdfA2A` or `PdfA3A`) and the copy option `CopyLogicalStructure` is not set.

`IllegalState`

- The document has already been closed.
- The `CopyFormFields` flag is set in the `options` parameter and the document contains form fields that have either been explicitly copied with `CopyFormFieldNode` or created with `CreateCheckBoxField`, `CreateComboBoxField`, `CreateCombTextField`, `CreateGeneralTextField`, `CreateListBoxField`, `CreateRadioButtonField`, or `CreateSubForm`.
- The `CopyOutlines` is set in the `options` argument and the output document contains outline items that have been explicitly copied with `CopyOutlineItem`.

`UnsupportedOperation`

- The document is read-only.

- The `page` argument is not associated with an input document.

`IllegalArgument`

- The `page` argument is `null`.
- The `page` argument has already been closed.

6.20.13 CopyPageAsGroup

Method: `Group CopyPageAsGroup(Page page, CopyOption options)`
Error codes: `IO`, `Corrupt`, `Conformance`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Assembly`

Create a group object from a page of a different document.

A group that contains interactive elements can be painted once only. Interactive elements are annotations, form fields, outlines or logical structure information. If a group needs to be painted multiple times, interactive elements can be flattened or the group can be copied multiple times from the page.

There are some interactive elements such as form fields or text annotations that cannot be rotated. So if you plan to rotate the group, it is recommended to flatten form fields and annotations.

Thread safety: This method needs **exclusive** access to the source document.

Parameters:

`page` `[Page]` A page of a different document.

`options` `[CopyOption]` The copy options.

Returns:

The copied group, associated with the current document.

Error Codes:

`IO` Error reading from the source document or writing to the target document.

`Corrupt` The source document is corrupt.

`Conformance`

- The conformance level of the source document is not compatible with the conformance level of the target document.
- The explicitly requested conformance level is PDF/A Level A (`PdfA1A`, `PdfA2A` or `PdfA3A`) and the copy option `CopyLogicalStructure` is not set.

IllegalState

- The document has already been closed.
- The CopyFormFields flag is set in the options parameter and the document contains form fields that have either been explicitly copied with CopyFormFieldNode or created with CreateCheckBoxField, CreateComboBoxField, CreateCombTextField, CreateGeneralTextField, CreateListBoxField, CreateRadioButtonField, or CreateSubForm.
- The CopyOutlines is set in the options argument and the output document contains outline items that have been explicitly copied with CopyOutlineItem.

UnsupportedOperation

- The document is read-only.
- The page argument is not associated with an input document.

IllegalArgument

- The page argument is null.
- The page argument has already been closed.

6.20.14 CopyViewerSettings

Method: ViewerSettings CopyViewerSettings(ViewerSettings viewerSettings)
Error codes: IllegalState, UnsupportedOperation, IllegalArgument, Conformance

Copy viewer settings from a different document.

The newly created viewer settings are associated with the document, but not yet used as the document's viewer settings.

Parameter:

viewerSettings [ViewerSettings] A viewer settings object of a different document.

Returns:

The copied viewer settings, associated with the current document.

Error Codes:

IllegalState The document has already been closed.

UnsupportedOperation

- The document is read-only.
- The viewerSettings argument is not associated with an input document.

IllegalArgument

- The viewerSettings argument is null.
- The viewerSettings argument has already been closed.

`Conformance`

- The conformance level of the source document is not compatible with the conformance level of the target document.

6.20.15 Create

Method: `Document Create(Stream stream, Compliance conformance, EnycryptionParams encryption)`
Static
Error codes: `IO`, `License`, `IllegalArgument`
License features: `Annotation`, `Assembly`, `Creation`, `Forms`, `Modification`

Create a new PDF document.

Documents with created with this method are writable and can be modified.

Parameters:

`stream` `[Stream]` The stream where the PDF document is stored.

Both, read and write access is required.

`conformance` `[Compliance]`

The required conformance level of the PDF document. Adding pages or content from incompatible documents or using incompatible features will lead to a conformance error.

Using `Unknown`, the conformance is determined automatically, based on the conformance of the input documents and the requirements of the used features.

Note that for PDF/A document it is highly recommended to set an output intent using the `OutputIntent` property.

`encryption` `[EnycryptionParams]` The optional encryption parameters.

Returns:

The newly created document instance.

Error Codes:

`IO` Error writing to the stream.

`License` The product is not properly licensed.

`IllegalArgument` The `stream` argument is `null`.

6.20.16 CreateCheckBoxField

Method: `Field CreateCheckBoxField()`
Error codes: `IllegalState`, `UnsupportedOperation`
License feature: `Forms`

Create a new `CheckBoxField`.

Returns:

The newly created check box field.

Error Codes:

`IllegalState`

- The document has already been closed.
- The document contains form fields that have been implicitly copied by `CopyPage` with activated `CopyOption` `CopyFormFields`.

`UnsupportedOperation` The document is read-only.

6.20.17 CreateCircleAnnotation

Method: `CircleAnnotation CreateCircleAnnotation(Rectangle rectangle, StrokeParams stroke, Paint fillPaint)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Annotation`

Create a circle annotation.

The returned circle annotation is not yet part of any page. It can be added to a page's list of annotations.

Parameters:

`rectangle` `[Rectangle]` The location on the page.

`stroke` `[StrokeParams]` The paint used for stroking the circle. If `null` then the circle is not stroked. If `null` and `fillPaint` is also `null`, then a default paint is used.

`fillPaint` `[Paint]` The stroking parameters used for stroking the circle. The stroking paint is used as the annotation's main paint. If `null` then a default is used.

Returns:

The newly created circle annotation.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `rectangle` argument is `null`.
- The `stroke` argument's paint belongs to a different document
- The color space of the `stroke` argument's paint is not a device color space.
- The blend mode of the `stroke` argument's paint is other than `Normal`.
- The `stroke` argument's line cap style is other than `CapButt`.
- The `stroke` argument's line join style is other than `JoinMitter`.
- The `stroke` argument's miter limit is other than `10.0`.
- The `stroke` argument's dash phase is other than `0.0`.
- The `fill` argument belongs to a different document.
- The color space of the `fill` argument is not a device color space.
- The blend mode of the `fill` argument is other than `Normal`.

6.20.18 CreateComboBoxField

Method: `Field CreateComboBoxField()`
Error codes: `IllegalState`, `UnsupportedOperation`
License feature: `Forms`

Create a new `ComboBoxField`.

Returns:

The newly created combo box field.

Error Codes:

`IllegalState`

- The document has already been closed.
- The document contains form fields that have been implicitly copied by `CopyPage` with activated `CopyOption` `CopyFormFields`.

`UnsupportedOperation` The document is read-only.

6.20.19 CreateCombTextField

Method: `CombTextField CreateCombTextField(int maxLength)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Forms`

Create a new `CombTextField`.

Parameter:

`maxLength` `[int]` The maximal character length for this field.

Returns:

The newly created comb text field.

Error Codes:

`IllegalState`

- The document has already been closed.
- The document contains form fields that have been implicitly copied by `CopyPage` with activated `CopyOption` `CopyFormFields`.

`UnsupportedOperation` The document is read-only.

`IllegalArgument` The given `maxLength` is smaller than `0`.

6.20.20 `CreateCustomStampAnnotation`

Method: `CombTextField CreateCustomStampAnnotation(Rectangle rectangle)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Annotation`

Create a new `CustomStampAnnotation`.

The returned custom stamp annotation's `Appearance` has an empty content with size equal to the given `rectangle`, and with coordinate origin located at the bottom left corner. Use a `ContentGenerator` to generate the appearance's content prior to adding the stamp annotation to a page's list of annotations.

Parameter:

`rectangle` `[Rectangle]` The location of the annotation on the page.

Returns:

The newly created custom stamp annotation.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument` The given `rectangle` argument is `null`.

6.20.21 CreateDeviceColorSpace

Method: `ColorSpace CreateDeviceColorSpace(DeviceColorSpaceType type)`
Error codes: `IO`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Creation`

Get the device color space of a certain type.

If the conformance of the document was **explicitly** set to PDF/A and the document has no appropriate output intent, one of the following color spaces is created instead:

Gray A calibrated grayscale color space, based on sRGB.

RGB A calibrated RGB color space, based on sRGB.

CMYK An ICC based CMYK color space, based on an available CMYK color profile.

If the conformance of the document was **automatically determined** to be PDF/A and the document has no appropriate output intent, the document is downgraded to a plain PDF.

Parameter:

type [`DeviceColorSpaceType`] The color space type.

Returns:

The newly created color space object.

Error Codes:

IO Unable to read a required ICC profile or writing to the document.

IllegalState The document has already been closed.

UnsupportedOperation The document is read-only.

IllegalArgument The `type` argument is `null`.

6.20.22 CreateFileAttachmentAnnotation

Method: `FileAttachmentAnnotation CreateFileAttachmentAnnotation(Point location, FileReference attachedFile, Paint paint)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Annotation`

Create a file attachment annotation.

The returned file attachment annotation is not yet part of any page. It can be added to a page's list of annotations.

Parameters:

location [Point] The location of the annotation's upper left corner on the page.

attachedFile [FileReference] The file to be attached.

paint [Paint] This paint for the file attachment icon. If null then a default paint is used.

Returns:

The newly created file attachment annotation.

Error Codes:

IllegalState The document has already been closed.

UnsupportedOperation The document is read-only.

IllegalArgument

- The location argument is null.
- The color space of the paint argument is not a device color space.
- The blend mode of the paint argument is other than Normal.
- The paint argument belongs to a different document.
- The attachedFile argument is null.
- The attachedFile argument belongs to a different document.

6.20.23 CreateFileReference

Method: FileReference CreateFileReference(Stream data, String name, String mediaType, String description, Date modificationDate)
Error codes: IllegalState, UnsupportedOperation, IllegalArgument, IO, Conformance
License feature: Assembly

The newly created FileReference object belongs to the document but is not (yet) used as an embedded file. The object can be added to the list of embedded files or to the list of associated files.

Parameters:

data [Stream] A stream of the file to be added. Read access is required.

name [String] The name to be used for the embedded file. This name is presented to the user when viewing the list of embedded files.

mediaType [String] The mime type of the embedded file. Default: "application/octet-stream". Common values other than the default are "application/pdf", "application/xml", or "application/msword".

description [String] The description of the embedded file. This is presented to the user when viewing the list of embedded files.

modificationDate [Date] The modify date of the file. Default: current time.

Returns:

The newly created file reference object.

Error Codes:

IllegalState The document has already been closed.

UnsupportedOperation The document is read-only.

IllegalArgument

- The stream argument is null.
- The name argument is null or an empty string.

IO Error reading from the stream.

Conformance

- The document's conformance is PDF/A-1.
- The document's conformance is PDF/A-2 and the given data contains a file other than PDF/A-1 or PDF/A-2.

6.20.24 CreateFont

Method: Font CreateFont(Stream stream, bool embedded)
Error codes: IO, UnknownFormat, Corrupt, Conformance, IllegalState, UnsupportedOperation, IllegalArgument
License feature: Creation

Create a new font object from font file data.

Supported formats are:

- Type1
- CFF
- TrueType
- OpenType

Parameters:

stream [Stream] The font file data stream.

embedded [bool] true if the font shall be embedded in the document.

Note that this parameter must be true for PDF/A documents.

Returns:

The newly created Font object.

Error Codes:

`IO` Error reading from the font file or writing to the document.

`UnknownFormat` The font data has an unknown format.

`Corrupt` The font data is corrupt.

`Conformance` The `embedded` argument is `false` for PDF/A document.

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `stream` argument is `null`.
- The `family` argument is `null`.

6.20.25 CreateSystemFont

Method: `Font CreateSystemFont(String family, String style, bool embedded)`
Error codes: `NotFound`, `IO`, `UnknownFormat`, `Corrupt`, `Conformance`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Creation`

Create a new font object from an installed font.

Supported formats are:

- Type1
- CFF
- TrueType
- OpenType

Parameters:

`family` `[String]` The font family name (e.g. "Arial").

`style` `[String]` The font style (e.g. "Bold").

`embedded` `[bool]` `true` if the font shall be embedded in the document.

Note that this parameter must be `true` for PDF/A documents.

Returns:

The newly created Font object.

Error Codes:

`NotFound` There is no such font installed.

`IO` Error reading from the font file or writing to the document.

`UnknownFormat` The font data has an unknown format.

`Corrupt` The font data is corrupt.

`Conformance` The `embedded` argument is `false` for PDF/A document.

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument` The `family` argument is `null` or an empty string.

6.20.26 `CreateFreeDrawingAnnotation`

Method: `FreeDrawingAnnotation CreateFreeDrawingAnnotation(Path path, Paint paint)`
Error codes: `IllegalState`, `UnsupportedFeature`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Annotation`

Create a free drawing annotation.

The returned free drawing annotation is not yet part of any page. It can be added to a page's list of annotations.

Parameters:

`path` `[Path]` The path of the free drawing.

`paint` `[Paint]` This paint used for stroking the line. If `null` then a default paint is used.

Returns:

The newly created free drawing annotation.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedFeature` The document's conformance is PDF 2.0 and the `path` argument contains curve-to operations.

`UnsupportedOperation`

- The document is read-only.
- The `path` argument has not been constructed with a `PathGenerator`.
- The document's conformance is not PDF 2.0 and the `path` argument contains curve-to operations.
- The `path` argument contains rectangle operations.

IllegalArgument

- The path argument is null.
- The color space of the paint argument is not a device color space.
- The blend mode of the paint argument is other than Normal.
- The paint argument belongs to a different document.

6.20.27 CreateFreeTextAnnotation

Method: FreeTextAnnotation CreateFreeTextAnnotation(Rectangle rectangle, String content, Paint paint)
Error codes: IllegalState, UnsupportedOperation, IllegalArgument
License feature: Annotation

Create a free text annotation.

The returned free text annotation is not yet part of any page. It can be added to a page's list of annotations.

Parameters:

rectangle [Rectangle] The location on the page.

content [String] The text content.

paint [Paint] This paint used for the background of the free text annotation. If null then a default paint is used.

Returns:

The newly created free text annotation.

Error Codes:

IllegalState The document has already been closed.

UnsupportedOperation The document is read-only.

IllegalArgument

- The rectangle argument is null.
- The color space of the paint argument is not a device color space.
- The blend mode of the paint argument is other than Normal.
- The paint argument belongs to a different document.

6.20.28 CreateGeneralTextField

Method: `GeneralTextField CreateGeneralTextField()`
Error codes: `IllegalState`, `UnsupportedOperation`
License feature: `Forms`

Create a new `GeneralTextField`.

Returns:

The newly created general text form field object.

Error Codes:

`IllegalState`

- The document has already been closed.
- The document contains form fields that have been implicitly copied by `CopyPage` with activated `CopyOption` `CopyFormFields`.

`UnsupportedOperation` The document is read-only.

6.20.29 CreateGroup

Method: `Group CreateGroup(Size size)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License features: `Creation`, `Modification`

Create an empty group object.

Parameter:

`size` `[Size]` The size of the group.

Returns:

The newly created group object.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument` The `size` argument is `null`.

6.20.30 CreateICCColorSpace

Method: `ColorSpace CreateICCColorSpace(Stream stream)`
Error codes: `IO`, `UnknownFormat`, `Corrupt`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Creation`

Create an new color space object from an ICC color profile.

Parameter:

`stream` [`Stream`] The color profile data stream.

Returns:

The newly created color profile object.

Error Codes:

`IO` Error reading from the profile or writing to the document.

`UnknownFormat` The profile data has an unknown format.

`Corrupt` The profile data is corrupt.

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument` The `stream` argument is `null`.

6.20.31 CreateImage

Method: `Image CreateImage(Stream stream)`
Error codes: `IO`, `UnknownFormat`, `Corrupt`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Creation`

Create an image object from image data.

Supported formats are:

- BMP
- DIB
- JPEG
- JPEG2000
- JBIG2

- PNG
- GIF

Parameter:

`stream` [`Stream`] The image data stream.

Returns:

The newly created image object.

Error Codes:

`IO` Error reading from the image or writing to the document.

`UnknownFormat` The image data has an unknown format.

`Corrupt` The image data is corrupt.

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument` The `stream` argument is `null`.

6.20.32 `CreateImageMask`

Method: `ImageMask CreateImageMask(Stream stream)`
Error codes: `IO`, `UnknownFormat`, `Corrupt`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Creation`

Create an image mask object from image data.

Supported formats are:

- BMP
- DIB
- JBIG2
- PNG
- GIF

Parameter:

`stream` [`Stream`] The image data stream.

Returns:

The newly created image mask object.

Error Codes:

`IO` Error reading from the image or writing to the document.

`UnknownFormat` The image data has an unknown format or the format is not suitable for an image mask.

`Corrupt` The image data is corrupt.

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument` The `stream` argument is `null`.

6.20.33 `CreateLineAnnotation`

Method: `LineAnnotation CreateLineAnnotation(Point start, Point end, StrokeParams stroke)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Annotation`

Create a line annotation.

The returned line annotation is not yet part of any page. It can be added to a page's list of annotations.

Parameters:

`start` [`Point`] The line's start point.

`end` [`Point`] The line's end point.

`stroke` [`StrokeParams`] The stroking parameters used for stroking the line. The stroking paint is used as the annotation's main paint. If `null` then a default is used.

Returns:

The newly created line annotation.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `start` argument is `null`.
- The `end` argument is `null`.
- The `stroke` argument's paint belongs to a different document.
- The color space of the `stroke` argument's paint is not a device color space.
- The blend mode of the `stroke` argument's paint is other than `Normal`.
- The `stroke` argument's line cap style is other than `CapButt`.

- The `stroke` argument's line join style is other than `JoinMitter`.
- The `stroke` argument's miter limit is other than `10.0`.
- The `stroke` argument's dash phase is other than `0.0`.

6.20.34 CreateListBoxField

Method: `Field CreateListBoxField()`
Error codes: `IllegalState`, `UnsupportedOperation`
License feature: `Forms`

Create a new `ListBoxField`.

Returns:

The newly created list box field.

Error Codes:

`IllegalState`

- The document has already been closed.
- The document contains form fields that have been implicitly copied by `CopyPage` with activated `CopyOption` `CopyFormFields`.

`UnsupportedOperation` The document is read-only.

6.20.35 CreateMetadata

Method: `Metadata CreateMetadata(Stream xmp)`
Error codes: `IO`, `Corrupt`, `IllegalState`, `UnsupportedOperation`
License features: `Creation`, `Assembly`

Create a new metadata object.

The newly created metadata object is associated with the document but not (yet) used as the document metadata. The object can be use as document metadata using the `Metadata` property or as page metadata using the Metadata property of the page interface.

Parameter:

`xmp` `[Stream]` A stream containing an XMP file or `null` to create an empty metadata object.

Returns:

The newly created metadata object.

Error Codes:

`IO` Error reading from the XMP stream or writing to the target document.

`Corrupt` The XMP stream is corrupt.

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

6.20.36 CreateNamedDestination

Method: `NamedDestination CreateNamedDestination(String name, DirectDestination target)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License features: `Creation`, `Assembly`

Create a new named destination object.

Parameters:

`name` `[String]` The name by which the destination is referred to.

`target` `[DirectDestination]` The target destination.

Returns:

The newly created named destination object.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `name` argument is `null`.
- The `target` argument is `null`.
- The `target` argument has already been closed.
- The `target` argument points to a page in a different document.

6.20.37 CreateOutlineItem

Method: `OutlineItem CreateOutlineItem(String title, Destination destination)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License features: `Creation`, `Assembly`

Create a new outline item (bookmark).

The returned outline item is not yet part of the outline item tree. It can be inserted at any position in the tree.

Parameters:

`title` [`String`] The title of the newly created outline item.

`destination` [`Destination`] The destination that this outline item refers to or `null` if the item has no destination.

Returns:

The newly created outline item object.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `title` argument is `null`.
- The `destination` argument has already been closed.
- The `destination` argument points to a page in a different document.

6.20.38 CreatePage

Method: `Page CreatePage(Size size)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License features: `Assembly`, `Creation`, `Forms`, `Modification`

Create an empty page.

The page is associated with the document but not yet part of it. It can be appended to the page list (see: `Pages`).

Parameter:

`size` [`Size`] The page size.

Returns:

The newly created page object.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

IllegalArgument The size argument is null.

6.20.39 CreatePolyLineAnnotation

Method: PolyLineAnnotation CreatePolyLineAnnotation(Path path, StrokeParams stroke)
Error codes: IllegalState, UnsupportedFeature, UnsupportedOperation, IllegalArgument, Conformance
License feature: Annotation

Create a poly line annotation.

The returned poly line annotation is not yet part of any page. It can be added to a page's list of annotations.

Parameters:

path [Path] The path of the poly line.

stroke [StrokeParams] The stroking parameters used for stroking the poly-line. The stroking paint is used as the annotation's main paint. If null then a default is used.

Returns:

The newly created poly line annotation.

Error Codes:

IllegalState The document has already been closed.

UnsupportedFeature The document's conformance is PDF 2.0 and the path argument contains curve-to operations.

UnsupportedOperation

- The document is read-only.
- The path argument has not been constructed with a PathGenerator.
- The document's conformance is not PDF 2.0 and the path argument contains curve-to operations.
- The path argument contains close-subpath or rectangle operations.

IllegalArgument

- The path argument is null.
- The stroke argument's paint belongs to a different document.
- The color space of the stroke argument's paint is not a device color space.
- The blend mode of the stroke argument's paint is other than Normal.
- The stroke argument's line cap style is other than CapButt.
- The stroke argument's line join style is other than JoinMitter.
- The stroke argument's miter limit is other than 10.0.
- The stroke argument's dash phase is other than 0.0.

Conformance The explicitly requested PDF conformance is smaller than PDF 1.5.

6.20.40 CreatePolygonAnnotation

Method: `PolygonAnnotation CreatePolygonAnnotation(Path path, StrokeParams stroke, Paint fillPaint)`
Error codes: `IllegalState`, `UnsupportedFeature`, `UnsupportedOperation`, `IllegalArgument`, `Conformance`
License feature: `Annotation`

Create a polygon annotation.

The returned polygon annotation is not yet part of any page. It can be added to a page's list of annotations.

Parameters:

`path` [`Path`] The path of the polygon.

`stroke` [`StrokeParams`] The stroking parameters used for stroking the polygon. The stroking paint is used as the annotation's main paint. If `null` then a default is used.

`fillPaint` [`Paint`] The paint used for filling the polygon. If `null` then the polygon is not filled.

Returns:

The newly created polygon annotation.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedFeature` The document's conformance is PDF 2.0 and the `path` argument contains curve-to operations.

`UnsupportedOperation`

- The document is read-only.
- The `path` argument has not been constructed with a `PathGenerator`.
- The document's conformance is not PDF 2.0 and the `path` argument contains curve-to operations.
- The `path` argument contains more than one subpath.
- The `path` argument's subpath has not been closed.

`IllegalArgument`

- The `path` argument is `null`.
- The `stroke` argument's paint belongs to a different document.
- The color space of the `stroke` argument's paint is not a device color space.
- The blend mode of the `stroke` argument's paint is other than `Normal`.
- The `stroke` argument's line cap style is other than `CapButt`.
- The `stroke` argument's line join style is other than `JoinMitter`.
- The `stroke` argument's miter limit is other than `10.0`.
- The `stroke` argument's dash phase is other than `0.0`.
- The `fill` argument belongs to a different document.

- The color space of the `fill` argument is not a device color space.
- The blend mode of the `fill` argument is other than `Normal`.

`Conformance` The explicitly requested PDF conformance is smaller than PDF 1.5.

6.20.41 `CreateRadioButtonField`

Method: `Field CreateRadioButtonField()`
Error codes: `IllegalState`, `UnsupportedOperation`
License feature: `Forms`

Create a new `RadioButtonField`.

Returns:

The newly created radio button field.

Error Codes:

`IllegalState`

- The document has already been closed.
- The document contains form fields that have been implicitly copied by `CopyPage` with activated `CopyOption` `CopyFormFields`.

`UnsupportedOperation` The document is read-only.

6.20.42 `CreateSolidPaint`

Method: `Paint CreateSolidPaint(ColorSpace colorSpace, double[] color)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Creation`

Create an new solid paint in a color space.

Parameters:

`colorSpace` `[ColorSpace]` The color space of the paint.

`color` `[double[]]` The color components. The allowed value range is dependent on the color space, for most color spaces it is 0.0 to 1.0.

Returns:

The newly created paint object.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `colorSpace` argument is `null`.
- The `colorSpace` argument has already been closed.
- The `colorSpace` argument is associated with a different document.
- The `color` argument is `null`.
- The `color` argument contains too few elements.
- An element of the `color` argument is out of range.

6.20.43 CreateAlphaPaint

Method: `Paint CreateAlphaPaint(ColorSpace colorSpace, double alpha, double[] color)`
Error codes: `Conformance`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Creation`

Create an new transparent paint in the specified color space.

Parameters:

`colorSpace` `[ColorSpace]` The color space of the paint.

`alpha` `[double]` The opacity (alpha) of the paint.

`color` `[double[]]` The color components. The allowed value range is dependent on the color space, for most color spaces it is 0.0 to 1.0.

Returns:

The newly created paint object.

Error Codes:

`Conformance` The `alpha` argument is not 1.0 and the explicitly specified conformance does not support transparency (PDF/A-1, PDF 1.0 - 1.3).

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `colorSpace` argument is `null`.
- The `colorSpace` argument has already been closed.
- The `colorSpace` argument is associated with a different document.

- The `alpha` argument is not in the allowed range of 0.0 to 1.0
- The `color` argument is `null`.
- The `color` argument contains too few elements.
- An element of the `color` argument is out of range.

6.20.44 CreateBlendingPaint

Method: `Paint CreateBlendingPaint(ColorSpace colorSpace, BlendMode blendMode, double alpha, double[] color)`
Error codes: `Conformance`, `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Creation`

Create an new transparent paint with a blend mode.

Parameters:

`colorSpace` `[ColorSpace]` The color space of the paint.

`blendMode` `[BlendMode]` The blend mode of the paint.

`alpha` `[double]` The opacity (alpha) of the paint.

`color` `[double[]]` The color components. The allowed value range is dependent on the color space, for most color spaces it is 0.0 to 1.0.

Returns:

The newly created paint object.

Error Codes:

`Conformance`

- The `alpha` argument is not 1.0 and the explicitly specified conformance does not support transparency (PDF/A-1, PDF 1.0 - 1.3).
- The `blendMode` argument is not `Normal` and the explicitly specified conformance does not support tranparency (PDF/A-1, PDF 1.0 - 1.3).

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `colorSpace` argument is `null`.
- The `colorSpace` argument has already been closed.
- The `colorSpace` argument is associated with a different document.
- The `alpha` argument is not in the allowed range of 0.0 to 1.0
- The `color` argument is `null`.
- The `color` argument contains too few elements.
- An element of the `color` argument is out of range.

6.20.45 CreateStickyNoteAnnotation

Method: `StickyNoteAnnotation CreateStickyNoteAnnotation(Point location, String content, Paint paint)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Annotation`

Create a sticky note annotation.

The returned sticky note annotation is not yet part of any page. It can be added to a page's list of annotations.

Parameters:

`location` [`Point`] The location of the annotation's upper left corner on the page.

`content` [`String`] The text content.

`paint` [`Paint`] This paint for the sticky note icon. If `null` then a default paint is used.

Returns:

The newly created sticky note annotation.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `location` argument is `null`.
- The color space of the `paint` argument is not a device color space.
- The blend mode of the `paint` argument is other than `Normal`.
- The `paint` argument belongs to a different document.

6.20.46 CreateSubForm

Method: `SubForm CreateSubForm()`
Error codes: `IllegalState`, `UnsupportedOperation`
License feature: `Forms`

Create a new `SubForm`.

Returns:

Error Codes:

`IllegalState`

- The document has already been closed.
- The document contains form fields that have been implicitly copied by `CopyPage` with activated `CopyOption` `CopyFormFields`.

`UnsupportedOperation` The document is read-only.

6.20.47 `CreateText`

Method: `Text CreateText()`
Error code: `IllegalState`
License feature: `Creation`

Create a new text object that is associated with this document.

Returns:

The newly created text object.

Error Code:

`IllegalState` The document has already been closed.

6.20.48 `CreateSquareAnnotation`

Method: `SquareAnnotation CreateSquareAnnotation(Rectangle rectangle, StrokeParams stroke, Paint fill)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Annotation`

Create a square annotation.

The returned square annotation is not yet part of any page. It can be added to a page's list of annotations.

Parameters:

`rectangle` `[Rectangle]` The location on the page.

`stroke` `[StrokeParams]` The stroking parameters used for stroking the square. The stroking paint is used as the annotation's main paint. If `null` then a default is used.

`fill` `[Paint]` The paint used for filling the square. If `null` then the square is not filled.

Returns:

The newly created square annotation.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `rectangle` argument is `null`
- The `stroke` argument's paint belongs to a different document.
- The color space of the `stroke` argument's paint is not a device color space.
- The blend mode of the `stroke` argument's paint is other than `Normal`.
- The `stroke` argument's line cap style is other than `CapButt`.
- The `stroke` argument's line join style is other than `JoinMitter`.
- The `stroke` argument's miter limit is other than `10.0`.
- The `stroke` argument's dash phase is other than `0.0`.
- The `fill` argument belongs to a different document.
- The color space of the `fill` argument is not a device color space.
- The blend mode of the `fill` argument is other than `Normal`.

6.20.49 CreateTextStampAnnotation

Method: `TextStampAnnotation CreateTextStampAnnotation(Point location, double? height, TextStampType textType, String text)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Annotation`

Create a text stamp annotation.

The width of the annotation is computed from the given stamp text. The returned text stamp annotation is not yet part of any page. It can be added to a page's list of annotations.

Parameters:

`location` [`Point`] The location of the annotation's upper left corner on the page.

`height` [`double?`] The height of the text stamp.

`textType` [`TextStampType`] The text stamp type.

`text` [`String`] The text to be shown in the stamp.

Returns:

The newly created text stamp annotation.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The `location` argument is `null`.
- The `textType` argument is `CustomTextStamp`.
- The `text` argument is `null` or empty.

6.20.50 EmbeddedFiles

Property (get): `FileReferenceList EmbeddedFiles`
Error code (get): `IllegalState`
License features: `Assembly`, `Extraction`

This is the list of the embedded files of this document.

If the document is writable, then it is possible to append new `FileReference`s to the list. Every `FileReference` object can occur at most once in this list.

For PDF/A-3 documents, appending to this list is not recommended and, if necessary, should be done after having copied `AssociatedFiles` and pages including associated files.

For PDF 2.0 documents, appending to this list should be done after having copied `AssociatedFiles` and pages including associated files.

Appending to this list results in a `Conformance` error if

- the document's `Conformance` is PDF/A-1. (PDF/A-1 does not support embedded files.)
- the document's `Conformance` is PDF/A-2 and the `FileReference` to be added contains a file other than a PDF/A-1 or PDF/A-2. (PDF/A-2 does not support other types of embedded files.)
- the document's `Conformance` is PDF/A-3 and the `FileReference` to be added is not associated to the document or to a page. (In PDF/A-3, all embedded files must be associated.)

Error Code:

`IllegalState` The document has already been closed.

6.20.51 FormFields

Property (get): `FormFieldNodeMap FormFields`
Error code (get): `IllegalState`
License features: `Extraction`, `Forms`

This map contains all AcroForm form fields that belong to this document.

Adding to this map can result in the following errors:

`IllegalState` if the map has already been closed

`UnsupportedOperation` if the document is read-only

`IllegalArgument`

- if the given `FormFieldNode` is `null`.
- if the given `FormFieldNode` has already been closed.
- if the given `FormFieldNode` does not belong to the same document as the map.
- if the given `FormFieldNode` has already been added to a form field node map.
- if the given key is `null`.
- if the given key contains a full stop ".".
- if the given key is already used in this map.

6.20.52 Metadata

Property (get, set): `Metadata Metadata`
Error codes (set): `IllegalState`, `UnsupportedOperation`, `IllegalArgument`

Get or set the metadata of the document.

If the document is writable, the metadata object will be writable too and all changes to the metadata object are reflected in the document. When setting the metadata the entire metadata of the document is replaced.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The metadata object is `null`.
- The metadata object belongs to a different document.
- The metadata object has already been closed.

6.20.53 Open

Method: `Document Open(Stream stream, String password)`
Static
Error codes: `Password`, `IO`, `Corrupt`, `UnsupportedFeature`, `Conformance`, `License`, `IllegalArgument`
License features: `Annotation`, `Assembly`, `Extraction`, `Forms`, `Modification`

Open a PDF document.

Documents opened with this method are read-only and cannot be modified.

Parameters:

`stream` [`Stream`] The stream where the PDF document is stored.

Read access is required.

`password` `[String]` The password to open the PDF document.

Returns:

The newly created document instance.

Error Codes:

`Password` The file is encrypted and the password is not valid.

`IO` Error reading from the stream.

`Corrupt` The file is corrupt or not a PDF.

`UnsupportedFeature`

- The file is a PDF collection.
- The PDF contains unrendered XFA fields.

`Conformance` The document's conformance level is not supported.

`License` The product is not properly licensed.

`IllegalArgument` The `stream` argument is `null`.

6.20.54 OpenDestination

Property (get, set): `Destination OpenDestination`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

The destination that is displayed when the document is opened.

This property corresponds to the `/OpenAction` entry in the document catalog. If the `/OpenAction` entry is neither a destination nor a `/GoTo` action, the value of this property is `null`. Action types other than `/GoTo` are not supported.

6.20.55 OutlineItems

Property (get): `OutlineItemList OutlineItems`
Error code (get): `IllegalState`

Get the document outline (also known as "Bookmarks").

This property may never be `null`.

Error Code:

`IllegalState` The document has already been closed.

6.20.56 `OutputIntent`

Property (get, set): `ColorSpace OutputIntent`
Error codes (set): `UnsupportedOperation`, `IllegalArgument`, `IllegalState`

Get or set the output intent of the document. If the document has no output intent, `null` is returned.

The output intent specifies a color profile that characterizes the intended output device. It is used to render device colors on devices other than the intended output device.

If the document's conformance is set to PDF/A and no output intent is set, all device colors are converted to calibrated colors while copying.

This method may not be called after any of the following methods have been called:

- `CopyPage`
- `CopyPageAsGroup`
- `CreatePage`
- `CreateGroup`

The color space used as output intent. The following restrictions apply for the color space used as output intent:

- The color space has to be ICC profile based.
- The color space has to be associated with the document, either by `CopyColorSpace` or by `CreateICCColorSpace`.

Error Codes:

`UnsupportedOperation`

- The document is read-only.
- An output intent has been set already.

`IllegalArgument`

- The given argument is `null`.
- The given argument has already been closed.
- The given argument is not an ICC profile based color space.
- The given argument is associated with a different document.

`IllegalState` The document has already been closed.

6.20.57 Pages

Property (get): `PageList Pages`
Error code (get): `IllegalState`

Return the list of pages of the document.

If the document is writable, it is possible to append new pages to the end of the list.

Error Code:

`IllegalState` The document has already been closed.

6.20.58 Permissions

Property (get): `Permission? Permissions`
Error code (get): `IllegalState`

The permissions in force for this document.

This property is `null` if the document is not encrypted.

Error Code:

`IllegalState` The document has already been closed.

6.20.59 SignatureFields

Property (get): `SignatureFieldList SignatureFields`
Error code (get): `IllegalState`
License feature: `Extraction`

The list of signature fields of the document.

Signature fields are a special kind of form fields, that can contain digital signatures.

Error Code:

`IllegalState` The document has already been closed.

6.20.60 ViewerSettings

Property (get, set): `ViewerSettings ViewerSettings`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`, `IllegalArgument`

The settings to use when opening the document in a viewer.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The given argument is `null`.
- The given argument is associated with a different document.

6.21 `DrawingAnnotation` Interface

An annotation used to draw something on a page.

Hierarchy:

```
Annotation
└─ MarkupAnnotation
    └─ DrawingAnnotation
        ├─ LineAnnotation
        ├─ PolyDrawingAnnotation
        │   ├─ FreeDrawingAnnotation
        │   ├─ PolyLineAnnotation
        │   └─ PolygonAnnotation
        └─ ShapeDrawingAnnotation
            ├─ CircleAnnotation
            └─ SquareAnnotation
```

6.21.1 `Paint`

Property (get): `Paint Paint`
Error codes (get): `IllegalState`, `Corrupt`

The paint for line stroking and for the popup.

Error Codes:

`IllegalState` The `DrawingAnnotation` object or the owning document has already been closed.

`Corrupt` The PDF is corrupt and the annotation's paint cannot be read.

6.21.2 Popup

Property (get, set): `AnnotationPopup Popup`
Error code: `IllegalState`

The annotation's popup.

Error Code:

`IllegalState` The `DrawingAnnotation` object or the owning document has already been closed.

6.22 EmbeddedPdfLink Interface

A link annotation that links to an embedded PDF document.

Hierarchy:

```
Annotation
└─ Link
    └─ EmbeddedPdfLink
```

6.22.1 EmbeddedPdfLink Constructor

Method: `EmbeddedPdfLink(Rectangle rectangle, FileReference fileReference)`
Error codes: `IllegalArgument`, `IllegalState`
License feature: `Annotation`

Create a new link to an embedded PDF document.

The link is associated with the document but not yet part of any page. It can be added to a page's list of `Annotations`.

Parameters:

`rectangle` `[Rectangle]` The location on the page.

`fileReference` `[FileReference]` The embedded PDF file.

Error Codes:

`IllegalArgument`

- The `rectangle` argument is `null`.
- The `fileReference` argument is `null`.
- The `fileReference` argument or the containing document has already been closed.
- The `fileReference` argument does not contain a PDF document.

`IllegalState`

- The `fileReference` argument is neither used in a `FileAttachmentAnnotation`, nor has it been appended to the document's list of `EmbeddedFiles` or `AssociatedFiles`.
- The `fileReference` argument is used in a `FileAttachmentAnnotation` and this annotation has not been appended to a page's list of `Annotations`.

6.22.2 NewWindow

Property (get, set): `boolean? NewWindow`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`, `IllegalState`

This defines the viewer's behavior when opening the target PDF document.

`null` The viewer uses its default behavior.

`true` Open the document in an additional window.

`false` Replace the parent document with the embedded document.

Error Codes:

`IllegalState` The annotation has already been appended to a page's list of `Annotations`.

`UnsupportedOperation` The document is read-only.

6.23 FileAttachmentAnnotation Interface

A file attachment annotation.

Hierarchy:

```
Annotation
└─ MarkupAnnotation
    └─ FileAttachmentAnnotation
```

6.23.1 AttachedFile

Property (get, set): `FileReference AttachedFile`
Error code: `IllegalState`

The embedded file.

Error Code:

`IllegalState` The `FileAttachmentAnnotation` object or the owning document has already been closed.

6.23.2 Icon

Property (get, set): `FileAttachmentIcon Icon`
Error code: `IllegalState`

The displayed icon.

Error Code:

`IllegalState` The `FileAttachmentAnnotation` object or the owning document has already been closed.

6.23.3 Paint

Property (get): `Paint Paint`
Error codes (get): `IllegalState`, `Corrupt`

The paint for the icon an the popup.

Error Codes:

`IllegalState` The `FileAttachmentAnnotation` object or the owning document has already been closed.

`Corrupt` The PDF is corrupt and the annotation's paint cannot be read.

6.23.4 Popup

Property (get, set): `AnnotationPopup Popup`
Error code: `IllegalState`

The annotation's popup.

Error Code:

`IllegalState` The `FileAttachmentAnnotation` object or the owning document has already been closed.

6.24 FileReference Interface

6.24.1 AssociationRelationship

Property (get, set): `String AssociationRelationship`
Error code: `IllegalState`

The file's association relationship.

This property is `null` if the file is not associated with any object. When associating a file reference with an object such as the document or a page, then this property defines the relationship between the file and the object.

Typical values are:

`"Source"` Used if this file is the original source material for the associated content.

`"Data"` Used if this file represents information used to derive a visual presentation such as for a table or a graph.

`"Alternative` Used if this file is an alternative representation of content, for example audio.

`"Supplement"` Used if this file represents a supplemental representation of the original source or data that may be more easily consumable (e.g., a MathML version of an equation).

`"EncryptedPayload"` Used if this file is an encrypted payload document that should be displayed to the user if the PDF processor has the cryptographic filter needed to decrypt the document.

`"FormData"` Used if this file is the data associated with form fields of this PDF.

`"Schema"` Used if this file is a schema definition for the associated object.

`"Unspecified"` Used when the relationship is not known.

Error Code:

`IllegalState` The `FileReference` object or the owning document has already been closed.

6.24.2 Description

Property (get): `String Description`
Error code (get): `IllegalState`

The file's description.

For embedded files, this is the description of the file presented to the user in the list of embedded files.

Error Code:

`IllegalState` The `FileReference` object or the owning document has already been closed.

6.24.3 MediaType

Property (get): `String MediaType`
Error code (get): `IllegalState`

The file's MIME type.

Error Code:

`IllegalState` The `FileReference` object or the owning document has already been closed.

6.24.4 ModificationDate

Property (get): `Date ModificationDate`
Error code (get): `IllegalState`

The file's date of last modification.

Error Code:

`IllegalState` The `FileReference` object or the owning document has already been closed.

6.24.5 Name

Property (get): `String Name`
Error code (get): `IllegalState`

The file name.

For embedded files, this is the name presented to the user in a the list of embedded files.

Error Code:

`IllegalState` The `FileReference` object or the owning document has already been closed.

6.24.6 Data

Property (get): `Stream Data`
Error code (get): `IllegalState`
License feature: `Extraction`

The file's stream.

Error Code:

IllegalState The FileReference object or the owning document has already been closed.

6.25 FileReferenceList Interface

See Programming Interfaces for more information on how lists are modeled and used in the different interfaces the 3-Heights® PDF Toolbox API provides.

A list of FileReferences can either be a list of document's embedded files (EmbeddedFiles) or a list of associated files (AssociatedFiles).

6.26 FitHeightDestination Interface

A destination that fits the height of a page into the viewport

> **Note:** Many PDF viewers support different viewing modes like "fit page" or "fit width". A FitHeightDestination will change the current viewing mode to "fit height" in those viewers.
> Changing the viewing mode is usually not very well received by users and thus a LocationZoomDestination should be preferred in most cases.

Hierarchy:

```
Destination
 └─ DirectDestination
     └─ FitHeightDestination
```

6.26.1 FitHeightDestination Constructor

Method: FitHeightDestination(Page page, bool fitActualContent)
Error code: IllegalArgument
License features: Assembly, Creation

Create a new FitHeightDestination object.

Parameters:

page [Page] The page in the document that this destination is pointing to.

fitActualContent [bool] If true, the viewport is fitted to the height of the actual content of the page, instead of the height of the page.

See property FitActualContent for more information.

Error Code:

`IllegalArgument`

- The `page` argument is `null`.
- The `page` argument has already been closed.

6.26.2 `FitActualContent`

Property (get): `bool FitActualContent`
Error code (get): `IllegalState`

If `true`, the viewport is fitted to the height of the actual content of the page, instead of the height of the page.

> **Note:** Many PDF viewers simply ignore this property and always treat it as `false`, i.e. switching to "fit height" mode anyway.

Error Code:

`IllegalState`

- The destination has already been closed.
- The associated document has already been closed.

6.27 `FitPageDestination` Interface

A destination fits an entire page into the viewport.

> **Note:** Many PDF viewers support different viewing modes like "fit page" or "fit width". A `FitPageDestination` will change the current viewing mode to "fit page" in those viewers.
> Changing the viewing mode is usually not very well received by users and thus a `LocationZoomDestination` should be preferred in most cases.

Hierarchy:

```
Destination
 └─ DirectDestination
     └─ FitPageDestination
```

6.27.1 FitPageDestination Constructor

Method: `FitPageDestination(Page page, bool fitActualContent)`
Error code: `IllegalArgument`
License features: `Assembly`, `Creation`

Create a new FitPageDestination object.

Parameters:

page [Page] The page in the document that this destination is pointing to.

fitActualContent [bool] If `true`, the viewport is fitted to the height of the actual content of the page, instead of the height of the page.

See property `FitActualContent` for more information.

Error Code:

IllegalArgument

- The `page` argument is `null`.
- The `page` argument has already been closed.

6.27.2 FitActualContent

Property (get): `bool FitActualContent`
Error code (get): `IllegalState`

If `true`, the viewport is fitted to the actual content of the page, instead of the size of the page.

> **Note:** Many PDF viewers simply ignore this property and always treat it as `false`, i.e. switching to "fit page" mode anyway.

Error Code:

IllegalState

- The destination has already been closed.
- The associated document has already been closed.

6.28 FitRectangleDestination Interface

A destination that fits a specified area of a page into the viewport.

Note: Many PDF viewers support different viewing modes like “fit page” or “fit width”. A `FitRectangleDestination` will change the current viewing mode to standard mode in those viewers, i.e. deactivate any special mode like “fit page” or “fit width”.
Changing the viewing mode is usually not very well received by users and thus a `LocationZoomDestination` should be preferred in most cases.

Hierarchy:

```
Destination
 └─ DirectDestination
     └─ FitRectangleDestination
```

6.28.1 `FitRectangleDestination` Constructor

Method: `FitRectangleDestination(Page page, Rectangle rectangle)`
Error code: `IllegalArgument`
License features: `Assembly`, `Creation`

Create a new FitRectangleDestination object.

Parameters:

`page` `[Page]` The page in the document that this destination is pointing to.

`rectangle` `[Rectangle]` The rectangle that is displayed in the viewport.

See property `Zoom` for more information.

Error Code:

`IllegalArgument`

- The `page` argument is `null`.
- The `page` argument has already been closed.
- The `rectangle` argument is `null`.

6.28.2 `Rectangle`

Property (get): `Rectangle Rectangle`
Error code (get): `IllegalState`

The rectangle that is displayed in the viewport.

Error Code:

`IllegalState`

- The destination has already been closed.
- The associated document has already been closed.

6.29 `FitWidthDestination` Interface

A destination that fits the width of a page into the viewport

> **Note:** Many PDF viewers support different viewing modes like "fit page" or "fit width". A `FitWidthDestination` will change the current viewing mode to "fit width" in those viewers.
> Changing the viewing mode is usually not very well received by users and thus a `LocationZoomDestination` should be preferred in most cases.

Hierarchy:

```
Destination
 └─ DirectDestination
     └─ FitWidthDestination
```

6.29.1 `FitWidthDestination` Constructor

Method: `FitWidthDestination(Page page, bool fitActualContent)`
Error code: `IllegalArgument`
License features: `Assembly`, `Creation`

Create a new FitWidthDestination object.

Parameters:

`page` `[Page]` The page in the document that this destination is pointing to.

`fitActualContent` `[bool]` If `true`, the viewport is fitted to the height of the actual content of the page, instead of the height of the page.

See property `FitActualContent` for more information.

Error Code:

`IllegalArgument`

- The `page` argument is `null`.

- The page argument has already been closed.

6.29.2 FitActualContent

Property (get): `bool FitActualContent`
Error code (get): `IllegalState`

If `true`, the viewport is fitted to the width of the actual content of the page, instead of the width of the page.

> **Note:** Many PDF viewers simply ignore this property and always treat it as `false`, i.e. switching to "fit page" mode anyway.

Error Code:

`IllegalState`

- The destination has already been closed.
- The associated document has already been closed.

6.30 Font Interface

6.30.1 BaseFont

Property (get): `String BaseFont`
Error codes (get): `IllegalState`, `Corrupt`

Get the PostScript name of the font.

Error Codes:

`IllegalState` The `Font` object or the owning document has already been closed.

`Corrupt` The base font is missing or cannot be decoded.

6.30.2 ItalicAngle

Property (get): `double ItalicAngle`
Error code (get): `IllegalState`

Get the italic angle of the font.

The angle is expressed in degrees counterclockwise from the vertical, of the dominant vertical strokes of the font. (For example, the 9-o'clock position is 90°, and the 3-o'clock position is -90°.) The value is negative for fonts that slope to the right, as almost all italic fonts do.

If an error occurs (because the font or the owning document is already closed) this method returns `0`. But a return value of `0` is not generally an indication for failure.

Error Code:

`IllegalState` The `Font` object or the owning document has already been closed.

6.30.3 Ascent

Property (get): `double Ascent`
Error code (get): `IllegalState`

Get the ascent of the font.

the maximum height above the baseline reached by glyphs in this font, excluding the height of glyphs for accented characters.

If an error occurs (because the font or the owning document is already closed) this method returns `0`. But a return value of `0` is not generally an indication for failure.

Error Code:

`IllegalState` The `Font` object or the owning document has already been closed.

6.30.4 Descent

Property (get): `double Descent`
Error code (get): `IllegalState`

Get the descent of the font.

The maximum depth below the baseline reached by glyphs in this font. The value is a negative number.

If an error occurs (because the font or the owning document is already closed) this method returns `0`. But a return value of `0` is not generally an indication for failure.

Error Code:

`IllegalState` The `Font` object or the owning document has already been closed.

6.30.5 CapHeight

Property (get): `double CapHeight`
Error code (get): `IllegalState`

Get the cap height of the font.

The vertical coordinate of the top of flat capital letters, measured from the baseline.

If an error occurs (because the font or the owning document is already closed) this method returns `0`. But a return value of `0` is not generally an indication for failure.

Error Code:

`IllegalState` The `Font` object or the owning document has already been closed.

6.30.6 GetCharWidth

Method: `double GetCharWidth(int character)`
Error code: `IllegalState`

Get the width of a single glyph.

The width of a unicode character (in pt) relative to a font size of 1 pt.

If an error occurs (because the font or the owning document is already closed) this method returns `0`. But a return value of `0` is not generally an indication for failure.

Error Code:

`IllegalState` The `Font` object or the owning document has already been closed.

6.31 FormField Interface

A form field is a node in the tree of form fields that cannot have any subordinate child fields, also called "terminal" form field.

This is the base class of all terminal form field types.

Hierarchy:

```
FormFieldNode
└─ FormField
    ├─ CheckBoxField
    ├─ ChoiceField
    │   ├─ ComboBoxField
    │   └─ ListBoxField
    ├─ PushButtonField
    ├─ RadioButtonField
    └─ TextField
        ├─ CombTextField
        └─ GeneralTextField
```

6.31.1 DoNotExport

Property (get, set): `bool DoNotExport`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

Tells whether this field is exported when exporting the form fields.

Error Codes:

`IllegalState` The `FormField` or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.31.2 ReadOnly

Property (get, set): `bool ReadOnly`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

Flags this field as read-only when viewing the document.

Error Codes:

`IllegalState` The `FormField` or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.31.3 Required

Property (get, set): `bool Required`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

Flags this field as mandatory.

Error Codes:

`IllegalState` The `FormField` or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.31.4 Widgets

Property (get, set): `WidgetList Widgets`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

This form field's `Widget` annotations.

Error Codes:

`IllegalState` The `FormField` or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.31.5 AddNewWidget

Method: `Field AddNewWidget(Rectangle rectangle)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Creation`

This method creates a widget (visual manifestation) for the form field. The widget is automatically added to the field's `Widgets`, but not to any page. This method does not work for radio button form fields; use `AddNewWidget`.

Parameter:

`rectangle` `[Rectangle]` The widget's target rectangle on the page.

Returns:

The newly created form field widget.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation`

- The document is read-only.
- The document contains form fields that have been implicitly copied by `CopyPage` with activated `CopyOption` `CopyFormFields`.
- The form field is of type `RadioButtonField`.

`IllegalArgument` The `rectangle` argument is `null`.

6.32 FormFieldNode Interface

This is the base class of all form field classes and of sub forms.

Hierarchy:

```
FormFieldNode
├─ FormField
│  ├─ CheckBoxField
│  ├─ ChoiceField
│  │  ├─ ComboBoxField
│  │  └─ ListBoxField
│  ├─ PushButtonField
│  ├─ RadioButtonField
│  └─ TextField
│     ├─ CombTextField
│     └─ GeneralTextField
└─ SubForm
```

6.32.1 DisplayName

Property (get, set):	`String DisplayName`
Error code (get):	`IllegalState`
Error codes (set):	`IllegalState`, `UnsupportedOperation`

The display name is not directly visible, but a PDF viewer can display this name, e.g., in a tool tip.

Error Codes:

`IllegalState` The `FormFieldNode` or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.32.2 ExportName

Property (get, set): `String ExportName`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

The name of this field used when exporting.

Error Codes:

`IllegalState` The `FormFieldNode` or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.33 FormFieldNodeMap Interface

See chapter 5 for more information on how maps (dictionaries) are modeled.

This map contains `FormFieldNode`s.

6.33.1 Lookup

Method: `FormFieldNode Lookup(String identifierPath)`
Error codes: `IllegalState`, `NotFound`

Look up the form field node given by the identifier path within this form field node map.

Error Codes:

`IllegalState` The document has already been closed.

`NotFound` No form field node with the given identifier path exists.

6.34 FreeTextAnnotation Interface

An annotation that displays text

For a `FreeTextAnnotation`, the annotation's content is used as text to be displayed as the annotation's visual manifestation on the page.

Hierarchy:

```
Annotation
└─ MarkupAnnotation
    └─ FreeTextAnnotation
```

6.34.1 Alignment

Property (get, set): `TextAlignment Alignment`
Error code: `IllegalState`

The text alignment.

Error Code:

`IllegalState` The `FreeTextAnnotation` object or the owning document has already been closed.

6.34.2 Paint

Property (get): `Paint Paint`
Error codes (get): `IllegalState`, `Corrupt`

The paint for text background and the popup.

Error Codes:

`IllegalState` The `FreeTextAnnotation` object or the owning document has already been closed.

`Corrupt` The PDF is corrupt and the annotation's paint cannot be read.

6.35 FreeDrawingAnnotation Interface

A free-hand drawing annotation.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ DrawingAnnotation
         └─ PolyDrawingAnnotation
             └─ FreeDrawingAnnotation
```

6.36 GeneralTextField Interface

This interface represents an ordinary text field.

Hierarchy:

```
FormFieldNode
 └─ FormField
     └─ TextField
         └─ GeneralTextField
```

6.36.1 DoNotScroll

Property (get, set):	`bool DoNotScroll`
Error code (get):	`IllegalState`
Error codes (set):	`IllegalState`, `UnsupportedOperation`

Flags this text field to have a non-scrollable user interface.

Error Codes:

`IllegalState` The `GeneralTextField` or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.36.2 DoNotSpellCheck

Property (get, set):	`bool DoNotSpellCheck`
Error code (get):	`IllegalState`
Error codes (set):	`IllegalState`, `UnsupportedOperation`

Flags this text field to have user interface in which spell checking is disabled.

Error Codes:

`IllegalState` The `GeneralTextField` or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.36.3 Multiline

Property (get, set): `bool Multiline`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

Flags this text field to have a user interface that permits line wrapping.

Error Codes:

`IllegalState`

- (get, set) The `GeneralTextField` or the owning document has already been closed.
- (set) The `GeneralTextField` has `Widgets`.

`UnsupportedOperation` The document is read-only.

6.36.4 Password

Property (get, set): `bool Password`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

Flags this text field to have a user interface that is suitable for password entry. Usually, a PDF viewer displays asterisks (*) instead of the characters entered or contained.

Error Codes:

`IllegalState`

- (get, set) The `GeneralTextField` or the owning document has already been closed.
- (set) The `GeneralTextField` has `Widgets`.

`UnsupportedOperation` The document is read-only.

6.37 Group Interface

6.37.1 Size

Property (get, set): `Size Size`
Error codes: `IO`, `Corrupt`, `IllegalState`

Get the size of the group.

Error Codes:

`IO` Error reading from the document.

`Corrupt` The document is corrupt.

`IllegalState`

- The owning document has already been closed.
- The group has already been closed.

6.37.2 Content

Property (get, set):	`Content Content`
Error codes:	`IO`, `Corrupt`, `IllegalState`

Get the group content.

If the group is writable, the content object can be used to apply new content on the group.

Error Codes:

`IO` Error reading from the document.

`Corrupt` The document is corrupt.

`IllegalState`

- The owning document has already been closed.
- The group has already been closed.

6.37.3 Isolated

Property (get, set):	`boolean Isolated`
Error code:	`IllegalState`

Get or set the transparency isolation behavior.

Error Code:

`IllegalState`

- The group has already been painted.
- The owning document has already been closed.
- The group has already been closed.

6.37.4 Knockout

Property (get, set): `boolean Knockout`
Error code: `IllegalState`

Get or set the transparency knockout behavior.

Error Code:

`IllegalState`

- The group has already been painted.
- The owning document has already been closed.
- The group has already been closed.

6.38 GroupElement Interface

A `ContentExtractor` yields such an element when encountering a PDF group unless un-grouping is configured, see `Ungrouping`.

Hierarchy:

```
ContentElement
└─ GroupElement
```

6.38.1 Group

Property (get): `Group Group`
Error code (get): `IllegalState`

Use this property to access this element's group.

Error Code:

`IllegalState` The `GroupElement` object or the owning document has already been closed.

6.39 HighlightAnnotation Interface

A text highlighting annotation.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ TextMarkupAnnotation
         └─ HighlightAnnotation
```

6.40 Image Interface

6.40.1 Width

Property (get): `int Width`
Error code (get): `IllegalState`

The width of the image in samples (pixels).

Error Code:

`IllegalState` The `Image` object or the containing document has already been closed.

6.40.2 Height

Property (get): `int Height`
Error code (get): `IllegalState`

The height of the image in samples (pixels).

Error Code:

`IllegalState` The `Image` object or the containing document has already been closed.

6.40.3 BitsPerComponent

Property (get): `int BitsPerComponent`
Error code (get): `IllegalState`

The number of bits used to represent each color component. Only a single value may be specified; the number of bits is the same for all color components. Valid values are 1, 2, 4, and 8.

Error Code:

`IllegalState` The `Image` object or the containing document has already been closed.

6.40.4 ColorSpace

Property (get): `ColorSpace ColorSpace`
Error code (get): `IllegalState`

The color space in which image samples are specified.

Error Code:

`IllegalState` The `Image` object or the containing document has already been closed.

6.41 ImageElement Interface

A `ContentExtractor` yields such an element when encountering a PDF image.

Hierarchy:

```
ContentElement
└─ ImageElement
```

6.41.1 Image

Property (get): `Image Image`
Error code (get): `IllegalState`

Use this property to access this element's image.

Error Code:

`IllegalState` The `ImageElement` object or the containing document has already been closed.

6.42 ImageMask Interface

6.42.1 Width

Property (get): `int Width`
Error code (get): `IllegalState`

The width of the image mask in samples (pixels).

Error Code:

`IllegalState` The `ImageMask` object or the containing document has already been closed.

6.42.2 Height

Property (get): `int Height`
Error code (get): `IllegalState`

The height of the image mask in samples (pixels).

Error Code:

`IllegalState` The `ImageMask` object or the containing document has already been closed.

6.43 ImageMaskElement Interface

A `ContentExtractor` yields such an element when encountering a PDF image mask.

Hierarchy:

```
ContentElement
└─ ImageMaskElement
```

6.43.1 ImageMask

Property (get): `ImageMask ImageMask`
Error code (get): `IllegalState`

Use this property to access this element's image mask.

Error Code:

`IllegalState` The `ImageMaskElement` object or the containing document has already been closed.

6.43.2 Paint

Property (get): `Paint Paint`
Error code (get): `IllegalState`

Use this property to access the paint used for filling the image mask.

Error Code:

`IllegalState` The `ImageMaskElement` object or the containing document has already been closed.

6.44 InternalLink Interface

A link annotation for document-wide links.

Hierarchy:

```
Annotation
└─ Link
    └─ InternalLink
```

6.44.1 Destination

Property (get): `Destination Destination`
Error code (get): `IllegalState`

The link's target destination.

Error Code:

`IllegalState`

- The `InternalLink` object or the owning document has already been closed.
- The `InternalLink` object has no destination.

6.45 LineAnnotation Interface

An annotation used to draw a line on a page.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ DrawingAnnotation
         └─ LineAnnotation
```

6.45.1 Start

Property (get, set): `Point Start`
Error code: `IllegalState`

The line's starting point.

Error Code:

`IllegalState` The `LineAnnotation` object or the owning document has already been closed.

6.45.2 End

Property (get, set): `Point End`
Error code: `IllegalState`

The line's ending point.

Error Code:

`IllegalState` The `LineAnnotation` object or the owning document has already been closed.

6.45.3 StartStyle

Property (get, set): `AnnotationLineEnding StartStyle`
Error code: `IllegalState`

The starting point's style.

Error Code:

`IllegalState` The `LineAnnotation` object or the owning document has already been closed.

6.45.4 EndStyle

Property (get, set): `AnnotationLineEnding EndStyle`
Error code: `IllegalState`

The ending point's style.

Error Code:

`IllegalState` The `LineAnnotation` object or the owning document has already been closed.

6.45.5 LineEndingFill

Property (get, set): `Paint LineEndingFill`
Error code: `IllegalState`

The line ending filling paint, applying to both the starting and the ending point.

Error Code:

`IllegalState` The `LineAnnotation` object or the owning document has already been closed.

6.46 Link Interface

A link annotation.

Hierarchy:

```
Annotation
└─ Link
    ├─ EmbeddedPdfLink
    ├─ InternalLink
    └─ WebLink
```

6.46.1 BorderStyle

Property (set): `StrokeParams BorderStyle`
Error codes (set): `IllegalState`, `UnsupportedOperation`, `IllegalArgument`, `UnsupportedFeature`

The link's border.

This property defines if and how a rectangular border is drawn for the link.

The only color spaces allowed for the `StrokeParams'Paint` are DeviceRGB and CalRGB, because many PDF viewers don't support other color spaces.

Error Codes:

`IllegalState` The `Link` object or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The given `StrokeParams` is `null`.
- The given `StrokeParams` has a `DashPhase` differing from `0`.
- The given `StrokeParams` has a `MiterLimit` differing from `10`.
- The given `StrokeParams` has a `LineCapStyle` other than `Butt`.
- The given `StrokeParams` has a `LineJoinStyle` other than `Miter`.
- The given `StrokeParams'Paint` is not `null` and has a color space other than a device color space, or a calibrated color space.

`UnsupportedFeature` The given `StrokeParams'Paint` is not `null` and has a DeviceGray, DeviceCMYK, or a CalGray color space.

6.47 ListBoxField Interface

This interface represents a list box with items to choose from.

Hierarchy:

```
FormFieldNode
 └─ FormField
     └─ ChoiceField
         └─ ListBoxField
```

6.47.1 AllowMultiSelect

Property (get, set): `bool AllowMultiSelect`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

When `true`, then the user is allowed to select multiple items in this list box field.

Error Codes:

`IllegalState` The `ListBoxField` or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.47.2 ChosenItems

Property (get): `ChoiceItemList ChosenItems`
Error code (get): `IllegalState`

The list of selected `ChoiceItem`s.

Adding to this list can result in the following errors:

`IllegalState`

- if the list has already been closed.
- if the form field is marked as `ReadOnly`.
- if this list is not empty and the `ListBoxField` does not allow multiple items to be selected (`AllowMultiSelect`.
- if the choice field has `Widgets`.

`UnsupportedOperation` if the document is read-only.

`IllegalArgument`

- if the given `ChoiceItem` is `null`.
- if the given `ChoiceItem` has already been closed.
- if the given `ChoiceItem` is already present in this list.
- if the given `ChoiceItem` does not belong to the `ChoiceField`'s `Items`.

Removing items or clearing the list results in an `IllegalState` error if the list box field is marked as `ReadOnly`, or if it has `Widgets`.

Error Code:

`IllegalState` The `ListBoxField` or the owning document has already been closed.

6.48 LocationZoomDestination Interface

A destination that points to a specific location on the target page, using a specified zoom factor. The location is displayed in the top left corner of the viewport (if possible).

Hierarchy:

```
Destination
 └─ DirectDestination
     └─ LocationZoomDestination
```

6.48.1 LocationZoomDestination Constructor

Method: `LocationZoomDestination(Page page, double? left, double? top, double? zoom)`
Error code: `IllegalArgument`
License features: `Creation`, `Assembly`

Create a new LocationZoomDestination object.

Parameters:

`page` [`Page`] The page in the document that this destination is pointing to.

`left` [`double?`] The location of the page that is displayed at the left border of the viewport or `null`.

See property `Left` for more information.

`top` [`double?`] The location of the page that is displayed at the top of the viewport or `null`.

See property `Top` for more information.

`zoom` [`double?`] The zoom factor that is applied when jumping to the destination or `null`.

See property `Zoom` for more information.

Error Code:

`IllegalArgument`

- The `page` argument is `null`.
- The `page` argument has already been closed.

6.48.2 Left

Property (get): `double? Left`
Error code (get): `IllegalState`

The location of the page that is displayed at the left border of the viewport (if possible).

If the property is `null`, the value from before the jump is retained.

Note: Due to the current zoom factor, it is usually not possible for viewers to scroll as far to the right side, as would be necessary to place the location at the left corner of the viewport. However, viewers will ensure, that the location is at least visible.
In practice this means, that this value is mostly irrelevant.

Error Code:

`IllegalState`

- The destination has already been closed.
- The associated document has already been closed.

6.48.3 Top

Property (get): `double? Top`
Error code (get): `IllegalState`

The location of the page that is displayed at the top of the viewport (if possible).

If the property is `null`, the value from before the jump is retained.

Error Code:

`IllegalState`

- The destination has already been closed.
- The associated document has already been closed.

6.48.4 Zoom

Property (get): `double? Zoom`
Error code (get): `IllegalState`

The zoom factor that is applied when jumping to the destination.

A value of `null` means that the current zoom level is retained.

> **Note:** Many PDF viewers support different viewing modes like “fit page” or “fit width”.
> A `LocationZoomDestination` with a `Zoom` value of `null` will usually **not** change the current viewing mode in most viewers.
> For other `Zoom` values however, the viewer must switch to the standard mode, i.e. deactivate special modes like “fit page” or “fit width”.
> Changing the viewing mode is usually not very well received by users and thus using a `Zoom` value other than `null` is discouraged.

Error Code:

`IllegalState`

- The destination has already been closed.
- The associated document has already been closed.

6.49 `MarkupAnnotation` Interface

Markup annotations annotate a page’s content.

Hierarchy:

```
Annotation
└─ MarkupAnnotation
   ├─ DrawingAnnotation
   │  ├─ LineAnnotation
   │  ├─ PolyDrawingAnnotation
   │  │  ├─ FreeDrawingAnnotation
   │  │  ├─ PolyLineAnnotation
   │  │  └─ PolygonAnnotation
   │  └─ ShapeDrawingAnnotation
   │     ├─ CircleAnnotation
   │     └─ SquareAnnotation
   ├─ FileAttachmentAnnotation
   ├─ FreeTextAnnotation
   ├─ StampAnnotation
   │  ├─ CustomStampAnnotation
   │  └─ TextStampAnnotation
   ├─ StickyNoteAnnotation
   ├─ TextInsertAnnotation
   └─ TextMarkupAnnotation
      ├─ HighlightAnnotation
      ├─ SquigglyAnnotation
      ├─ StrikeThroughAnnotation
      └─ UnderlineAnnotation
```

6.49.1 Locked

Property (get, set): `bool Locked`
Error code: `IllegalState`

This property specifies whether the markup annotation can be modified. It does not restrict modification of the `Info`'s `Content`.

Error Code:

`IllegalState` The `MarkupAnnotation` object or the owning document has already been closed.

6.49.2 Info

Property (get, set): `MarkupInfo Info`
Error code: `IllegalState`

The information content of this markup annotation.

Error Code:

`IllegalState` The `MarkupAnnotation` object or the owning document has already been closed.

6.49.3 Replies

Property (get, set): `MarkInfoList Replies`
Error code: `IllegalState`

The replies to this markup annotation.

Error Code:

`IllegalState` The `MarkupAnnotation` object or the owning document has already been closed.

6.50 MarkupInfo Interface

A `MarkupInfo` object holds information contained in an `MarkupAnnotation` or in a reply to a markup annotation.

6.50.1 CreationDate

Property (get): `Date CreationDate`
Error code (get): `IllegalState`

The date of creation.

Error Code:

`IllegalState` The `MarkupInfo` object or the owning document has already been closed.

6.50.2 Locked

Property (get): `bool Locked`
Error code (get): `IllegalState`

This property determines whether the `Content` can be modified. It does not restrict the modification of other aspects of the annotation or its deletion.

Error Code:

`IllegalState` The `MarkupInfo` object or the owning document has already been closed.

6.50.3 Author

Property (get): `String Author`
Error code (get): `IllegalState`

The author.

Error Code:

`IllegalState` The `MarkupInfo` object or the owning document has already been closed.

6.50.4 Subject

Property (get): `String Subject`
Error code (get): `IllegalState`

The subject.

Error Code:

`IllegalState` The `MarkupInfo` object or the owning document has already been closed.

6.50.5 Content

Property (get): `String Content`
Error code (get): `IllegalState`

The information content.

Error Code:

`IllegalState` The `MarkupInfo` object or the owning document has already been closed.

6.51 MarkupInfoList Interface

See Programming Interfaces for more information on how lists are modeled and used in the different interfaces the 3-Heights® PDF Toolbox API provides.

6.52 Metadata Interface

Represents the metadata of a document or an object in a document.

For document level metadata, all changes are reflected in both, XMP metadata and document info dictionary depending on the conformance of the document.

6.52.1 Author

Property (get, set): `String Author`
Error code: `IllegalState`

The name of the person who created the document or resource.

This property corresponds to the `<dc:creator>` entry in the XMP metadata and to the `/Author` entry in the document information dictionary.

Error Code:

`IllegalState` The `Metadata` object or the owning document has already been closed.

6.52.2 CreationDate

Property (get, set): `Date CreationDate`
Error code: `IllegalState`

> **Note:** In the .NET interface `CreationDate` has type `DateTime` and is deprecated. Instead you should use `CreationDate2` which has type `DateTimeOffset` and hence includes time zone information.

The date and time the document or resource was originally created.

This property corresponds to the `<xmp:CreateDate>` entry in the XMP metadata and to the `/CreationDate` entry in the document information dictionary.

Error Code:

`IllegalState` The `Metadata` object or the owning document has already been closed.

6.52.3 Creator

Property (get, set): `String Creator`
Error code: `IllegalState`

The name of the first known tool used to create the document or resource.

This property corresponds to the `<xmp:CreatorTool>` entry in the XMP metadata and to the `/Creator` entry in the document information dictionary.

Error Code:

`IllegalState` The `Metadata` object or the owning document has already been closed.

6.52.4 CustomEntries

Property (get): `StringMap CustomEntries`
Error code (get): `IllegalState`

The custom entries in the document information dictionary.

The standard entries "Title", "Author", "Subject", "Keywords", "CreationDate", "ModDate", "Creator", "Producer" and "Trapped" are not included in the map. Any attempt to set a standard entry through this map will result in an error. To get or set standard entries use the corresponding properties instead.

> **Note:** The document information dictionary has been superseded by XMP metadata and is deprecated in PDF 2.0.

Error Code:

`IllegalState` The `Metadata` object or the owning document has already been closed.

6.52.5 Keywords

Property (get, set): `String Keywords`
Error code: `IllegalState`

Keywords associated with the document or resource.

Keywords can be separated by:

- carriage return / line feed
- comma
- semicolon
- tab
- double space

This property corresponds to the `<pdf:Keywords>` entry in the XMP metadata and to the `/Keywords` entry in the document information dictionary.

Setting this property also sets the XMP property `<dc:subject>` accordingly.

Error Code:

`IllegalState` The `Metadata` object or the owning document has already been closed.

6.52.6 ModificationDate

Property (get): `Date ModificationDate`
Error code (get): `IllegalState`

> **Note:** In the .NET interface `ModificationDate` has type `DateTime` and is deprecated. Instead you should use `ModificationDate2` which has type `DateTimeOffset` and hence includes time zone information.

The date and time the document or resource was most recently modified.

This property corresponds to the `<xmp:ModifyDate>` entry in the XMP metadata and to the `/ModDate` entry in the document information dictionary.

Error Code:

`IllegalState` The `Metadata` object or the owning document has already been closed.

6.52.7 Producer

Property (get, set): `String Producer`
Error code: `IllegalState`

If the document was converted to PDF from another format, the name of the PDF processor that converted it to PDF.

This property corresponds to the `<pdf:Producer>` entry in the XMP metadata and to the `/Producer` entry in the document information dictionary.

Error Code:

`IllegalState` The `Metadata` object or the owning document has already been closed.

6.52.8 Subject

Property (get, set): `String Subject`
Error code: `IllegalState`

The subject of the document or resource.

This property corresponds to the `<dc:description>` entry in the XMP metadata and to the `/Subject` entry in the document information dictionary.

Error Code:

`IllegalState` The `Metadata` object or the owning document has already been closed.

6.52.9 Title

Property (get, set): `String Title`
Error code: `IllegalState`

The title of the document or resource.

This property corresponds to the `<dc:title>` entry in the XMP metadata and to the `/Title` entry in the document information dictionary.

Error Code:

`IllegalState` The `Metadata` object or the owning document has already been closed.

6.52.10 Xmp

Property (get): `Stream Xmp`
Error code (get): `IllegalState`

The XMP metadata stream or `null` if there is none.

The stream is read-only. To set the XMP stream of a metadata object use the method Document.CreateMetadata instead.

Error Code:

`IllegalState` The `Metadata` object or the owning document has already been closed.

6.53 NamedDestination Interface

A destination that can be referred to by name.

Named destinations have two advantages compared to direct destinations:

- The name can be used in web links, e.g. http://www.example.com/document.pdf#destinationname
- If the target destination of a named destination is changed, all occurrences automatically point ot the new target.

Hierarchy:

```
Destination
└─ NamedDestination
```

6.53.1 Name

Property (get): `String Name`
Error code (get): `IllegalState`

The name by which the destination is referred to.

Error Code:

`IllegalState`

- The destination has already been closed.
- The associated document has already been closed.

6.54 `OutlineItem` Interface

An outline item represents an entry in the outline tree of the document. It is also known as "Bookmark".

6.54.1 `Title`

Property (get, set): `String Title`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

The title of the outline item.

Error Codes:

`IllegalState`

- The outline item has already been closed.
- The associated document has already been closed.

`UnsupportedOperation` The document is read-only.

6.54.2 `Bold`

Property (get, set): `bool Bold`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

If `true`, the outline item is displayed in bold font.

Error Codes:

`IllegalState`

- The outline item has already been closed.
- The associated document has already been closed.

`UnsupportedOperation` The document is read-only.

6.54.3 `Italic`

Property (get, set): `bool Italic`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

If `true`, the outline item is displayed in italic font.

Error Codes:

`IllegalState`

- The outline item has already been closed.
- The associated document has already been closed.

`UnsupportedOperation` The document is read-only.

6.54.4 Destination

Property (get, set): `Destination Destination`
Error codes: `IllegalState`, `UnsupportedOperation`

The destination of the outline item.

This property is `null` if:

- the outline item has no associated destination or action.
- the associated action does not correspond to a destination in the same document. (e.g. remote destination, web link, JavaScript action, ...)

Error Codes:

`IllegalState`

- The outline item has already been closed.
- The associated document has already been closed.

`UnsupportedOperation` The document is read-only.

6.54.5 IsOpen

Property (get, set): `bool IsOpen`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

- If `true`, the item is expanded.
- If `false`, the item is collapsed.

This is property is only meaningful if the item has child items.

Error Codes:

`IllegalState`

- The outline item has already been closed.
- The associated document has already been closed.

`UnsupportedOperation` The document is read-only.

6.54.6 Children

Property (get): `OutlineItemList Children`
Error code (get): `IllegalState`

The child items of this outline item.

Error Code:

`IllegalState`

- The outline item has already been closed.
- The associated document has already been closed.

6.55 OutlineItemList Interface

See Programming Interfaces for more information on how lists are modeled and used in the different interfaces the 3-Heights® PDF Toolbox API provides.

6.56 Page Interface

6.56.1 Crop

Method: `void Crop(Rectangle rect)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Assembly`

Crop the page to a specified rectangle.

> **Note:** All page-related coordinates are normalized to the crop box of the page. Cropping the page thus changes the coordinate system, rendering all previously extracted coordinates invalid.

Parameter:

`rect` `[Rectangle]` The rectangle to crop the page to.

Error Codes:

`IllegalState` The `Page` object or the owning document has already been closed.

UnsupportedOperation The page is read-only.

IllegalArgument The `rect` argument is `null`.

6.56.2 Rotate

Method: `void Rotate(Rotation rotation)`
Error codes: `IllegalState`, `UnsupportedOperation`, `IllegalArgument`
License feature: `Assembly`

Rotate the page by a multiple of 90°.

Parameter:

rotation [`Rotation`] The desired rotation.

Error Codes:

IllegalState The `Page` object or the owning document has already been closed.

UnsupportedOperation The page is read-only.

IllegalArgument The `rotation` argument is `null`.

6.56.3 Annotations

Property (get): `AnnotationList Annotations`
Error code (get): `IllegalState`
License features: `Annotation`, `Extraction`, `Forms`

This list contains the page's `Annotation`s.

Adding to this list can result in the following errors:

IllegalState if the list or the owning document has already been closed.

UnsupportedOperation if the owning document is read-only.

IllegalArgument

- if the given `Annotation` is `null`.
- if the given `Annotation` object has already been closed.
- if the given `Annotation` does not belong to the same document as this list.
- if the given `Annotation` is already associated with a page.

This list does not support removing or setting elements or clearing.

6.56.4 ArtBox

Property (get): `Rectangle ArtBox`
Error code (get): `IllegalState`

The art box of the page.

The art box (PDF 1.3) defines the extent of the page's meaningful content (including potential white-space) as intended by the page's creator. The default value is the page's crop box.

This property is `null` if the page contains no explicit art box.

Error Code:

`IllegalState` The `Page` object or the owning document has already been closed.

6.56.5 BleedBox

Property (get): `Rectangle BleedBox`
Error code (get): `IllegalState`

The bleed box of the page.

The bleed box (PDF 1.3) defines the region to which the contents of the page shall be clipped when output in a production environment. This may include any extra bleed area needed to accommodate the physical limitations of cutting, folding, and trimming equipment. The actual printed page may include printing marks that fall outside the bleed box. The default value is the page's crop box.

This property is `null` if the page contains no explicit bleed box.

Error Code:

`IllegalState` The `Page` object or the owning document has already been closed.

6.56.6 Content

Property (get): `Content Content`
Error code (get): `IllegalState`
License features: `Assembly`, `Creation`, `Extraction`, `Modification`

Get the page content.

If the page is writable, the content object can be used to apply new content on the page, for example overlays or underlays.

Error Code:

`IllegalState` The `Page` object or the owning document has already been closed.

6.56.7 MediaBox

Property (get): `Rectangle MediaBox`
Error code (get): `IllegalState`

The media box of the page.

The media box defines the boundaries of the physical medium on which the page is to be printed. It may include any extended area surrounding the finished page for bleed, printing marks, or other such purposes. It may also include areas close to the edges of the medium that cannot be marked because of physical limitations of the output device. Content falling outside this boundary may safely be discarded without affecting the meaning of the PDF file.

This property cannot be `null`.

Error Code:

`IllegalState` The `Page` object or the owning document has already been closed.

6.56.8 Metadata

Property (get, set): `Metadata Metadata`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`, `IllegalArgument`

Get or set the metadata of the page.

If the document is writable, the metadata object will be writable too and all changes to the metadata object are reflected in the document. When setting the metadata, the entire metadata of the page is replaced.

This property is `null` if the page has not metadata. If the property is set to `null`, the existing page metadata is removed.

Error Codes:

`IllegalState` The `Page` object or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- The metadata object belongs to a different document.
- The metadata object has already been closed.

6.56.9 Size

Property (get): `Size Size`
Error code (get): `IllegalState`

Get the visible size of the page (crop box).

The page size corresponds to the size of the crop box. Since all coordinates are normalized to the origin of the crop box, the normalized origin of the crop box is always (0,0) and thus only the size is required.

The crop box defines the region to which the contents of the page shall be clipped (cropped) when displayed or printed. Unlike the other boxes, the crop box has no defined meaning in terms of physical page geometry or intended use; it merely imposes clipping on the page contents. However, in the absence of additional information (such as imposition instructions specified in a JDF job ticket), the crop box determines how the page's contents shall be positioned on the output medium. The default value is the page's media box.

This property cannot be `null`.

Error Code:

`IllegalState` The `Page` object or the owning document has already been closed.

6.56.10 `TrimBox`

Property (get): `Rectangle TrimBox`
Error code (get): `IllegalState`

The trim box of the page.

The trim box (PDF 1.3) defines the intended dimensions of the finished page after trimming. It may be smaller than the media box to allow for production-related content, such as printing instructions, cut marks, or colour bars. The default value is the page's crop box.

This property is `null` if the page contains no explicit trim box.

Error Code:

`IllegalState` The `Page` object or the owning document has already been closed.

6.57 `PageList` Interface

See Programming Interfaces for more information on how lists are modeled and used in the different interfaces the 3-Heights® PDF Toolbox API provides.

6.58 `Paint` Interface

An object of this type is either created using the `Document` interface and meant to be passed as an argument for various methods of the `ContentGenerator` interface, in which case getting any of the `Paint`'s properties results in an `IllegalState` error.

Alternatively, such an object is part of the `FillParams` or `StrokeParams` of a `ContentElement` and in this case getting its properties is legal.

6.58.1 Color

Property (get): `double Color`
Error code (get): `IllegalState`

The array contains the color values ranging from `0.0` to `1.0`.

Error Code:

`IllegalState`

- If the paint has been created using any of the respective methods in the `Document` interface.
- The `Paint` object or the owning document has already been closed.

6.58.2 ColorSpace

Property (get): `ColorSpace ColorSpace`
Error code (get): `IllegalState`

This is the color space of the paint.

Error Code:

`IllegalState`

- If the paint has been created using any of the respective methods in the `Document` interface.
- The `Paint` object or the owning document has already been closed.

6.58.3 Transparency

Property (get): `TransparencyParams Transparency`
Error code (get): `IllegalState`

The transparency parameters of this paint or `null` if the paint is opaque.

Error Code:

`IllegalState`

- If the paint has been created using any of the respective methods in the `Document` interface.
- The `Paint` object or the owning document has already been closed.

6.59 Path Interface

6.59.1 Path Constructor

Method: `Path(InsideRule rule)`
Error code: `IllegalArgument`

Create a new path object.

Parameter:

`rule` [`InsideRule`] The inside rule.

Error Code:

`IllegalArgument`

- The `rule` argument is `null`.

6.60 PathElement Interface

A `ContentExtractor` yields such an element when encountering a PDF path.

Hierarchy:

```
ContentElement
└─ PathElement
```

6.60.1 AlignmentBox

Property (get): `Rectangle AlignmentBox`
Error code (get): `IllegalState`

This is a rectangle in untransformed coordinates that encompasses all points of the path. In contrast to the `BoundingBox`, the alignment may not encompass the path completely. E.g. if the path is stroked, then the stroked lines extend beyond the individual points of a line due to the line's width, join, and cap.

Error Code:

`IllegalState` The `PathElement` object or the owning document has already been closed.

6.60.2 Fill

Property (get): `FillParams Fill`
Error code (get): `IllegalState`

This specifies the path's parameters for filling or `null` if the path is not filled.

Error Code:

`IllegalState` The `PathElement` object or the owning document has already been closed.

6.60.3 Path

Property (get): `Path Path`
Error code (get): `IllegalState`

Use this property to access this element's path.

Error Code:

`IllegalState` The `PathElement` object or the owning document has already been closed.

6.60.4 Stroke

Property (get): `StrokeParams Stroke`
Error code (get): `IllegalState`

This specifies the path's parameters for stroking or `null` if the path is not stroked.

Error Code:

`IllegalState` The `PathElement` object or the owning document has already been closed.

6.61 PathGenerator Interface

6.61.1 PathGenerator Constructor

Method: `PathGenerator(Path path)`
Error code: `IllegalArgument`
License feature: `Creation`

Create a new content generator for appending to a path.

Parameter:

`path` [`Path`] The path object.

Error Code:

`IllegalArgument`

- The `path` argument is `null`.
- The `path` argument has already been closed.

6.61.2 AddArc

Method: `void AddArc(Rectangle rect, double alpha1, double alpha2)`
Error codes: `IllegalState`, `IllegalArgument`

Add an elliptical arc to the current path.

Parameters:

`rect` [`Rectangle`] The surrounding rectangle of the ellipse.

`alpha1` [`double`] The angle between the x-axis and the begin of the arc.

`alpha2` [`double`] The angle between the x-axis and the end of the arc.

Error Codes:

`IllegalState` The `ContentGenerator` object or the `Path` object has already been closed.

`IllegalArgument` The `rect` argument is `null`.

6.61.3 AddCircle

Method: `void AddCircle(Point center, double radius)`
Error codes: `IllegalState`, `IllegalArgument`

Append a circle to the current path as a complete subpath.

Parameters:

`center` [`Point`] The center of the circle.

`radius` [`double`] The radius of the circle.

Error Codes:

`IllegalState` The `ContentGenerator` object or the `Path` object has already been closed.

`IllegalArgument` The `center` argument is `null`.

6.61.4 AddEllipse

Method: `void AddEllipse(Rectangle rect)`
Error codes: `IllegalState`, `IllegalArgument`

Add an ellipse to the current path as a complete subpath.

Parameter:

`rect` [`Rectangle`] The surrounding rectangle of the ellipse.

Error Codes:

`IllegalState` The `ContentGenerator` object or the `Path` object has already been closed.

`IllegalArgument` The `rect` argument is `null`.

6.61.5 AddPie

Method: `void AddPie(Rectangle rect, double alpha1, double alpha2)`
Error codes: `IllegalState`, `IllegalArgument`

Add an elliptical piece of pie to the current path as a complete subpath.

Parameters:

`rect` [`Rectangle`] The surrounding rectangle of the ellipse.

`alpha1` [`double`] The angle between the x-axis and the begin of the arc.

`alpha2` [`double`] The angle between the x-axis and the end of the arc.

Error Codes:

`IllegalState` The `ContentGenerator` object or the `Path` object has already been closed.

`IllegalArgument` The `rect` argument is `null`.

6.61.6 AddRectangle

Method: `void AddRectangle(Rectangle rect)`
Error codes: `IllegalState`, `IllegalArgument`

Append a rectangle to the current path as a complete subpath.

Parameter:

`rect` [`Rectangle`] The rectangle to be added to the path.

Error Codes:

`IllegalState` The `ContentGenerator` object or the `Path` object has already been closed.

`IllegalArgument` The `rect` argument is `null`.

6.61.7 BezierTo

Method: `void BezierTo(Point ctrl1, Point ctrl2, Point target)`
Error codes: `IllegalState`, `IllegalArgument`

Draw a Bézier curve.

Append a cubic Bézier curve to the current path. The curve extends from the current point to the Target position, using Ctrl1 and Ctrl2 as the Bézier control points.

The current position is changed to the target position.

Parameters:

`ctrl1` [`Point`] The first Bézier control point.

`ctrl2` [`Point`] The second Bézier control point.

`target` [`Point`] The target coordinates.

Error Codes:

`IllegalState` The `ContentGenerator` object or the `Path` object has already been closed.

`IllegalArgument`

- The `target` argument is `null`.
- The `ctrl1` argument is `null`.
- The `ctrl2` argument is `null`.

6.61.8 Close

Method: `void Close()`
Error code: `IllegalState`

Close the Generator.

The generated object is not finalized until this method is called.

All native data structures that belong to the generator are freed.

After closing the generator, any call to a method of the generator will result in an `IllegalState` error.

Error Code:

`IllegalState` The generator has already been closed.

6.61.9 CloseSubpath

Method: `void CloseSubpath()`
Error code: `IllegalState`

Close the current subpath.

Close the current subpath by appending a straight line segment from the current point to the starting point of the subpath. This operator terminates the current subpath; appending another segment to the current path will begin a new subpath, even if the new segment begins at the endpoint reached by the `CloseSubpath` operation. If the current subpath is already closed, `CloseSubpath` does nothing.

Error Code:

`IllegalState` The `ContentGenerator` object or the `Path` object has already been closed.

6.61.10 LineTo

Method: `void LineTo(Point target)`
Error codes: `IllegalState`, `IllegalArgument`

Draw a line.

Append a straight line segment from the current point to the target coordinates.

The current position is changed to the target position.

Parameter:

`target` [`Point`] The target coordinates.

Error Codes:

IllegalState The ContentGenerator object or the Path object has already been closed.

IllegalArgument The target argument is null.

6.61.11 MoveTo

Method: void MoveTo(Point target)
Error codes: IllegalState, IllegalArgument

Move the current position.

Begin a new subpath by moving the current point to the specified coordinates, omitting any connecting line segment. If the previous path construction operator in the current path was also MoveTo, the new MoveTo overrides it; no vestige of the previous MoveTo operation remains in the path.

Parameter:

target [Point] The target coordinates.

Error Codes:

IllegalState The ContentGenerator object or the Path object has already been closed.

IllegalArgument The target argument is null.

6.62 PolyDrawingAnnotation Interface

A drawing annotation with multiple points.

Hierarchy:

```
Annotation
└─ MarkupAnnotation
    └─ DrawingAnnotation
        └─ PolyDrawingAnnotation
            ├─ FreeDrawingAnnotation
            ├─ PolyLineAnnotation
            └─ PolygonAnnotation
```

6.62.1 Path

Property (get, set): `Path Path`
Error code: `IllegalState`

The drawing path.

Error Code:

`IllegalState` The `PolyDrawingAnnotation` object or the owning document has already been closed.

6.63 PolyLineAnnotation Interface

An drawing annotation that consists of several line segments.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ DrawingAnnotation
         └─ PolyDrawingAnnotation
             └─ PolyLineAnnotation
```

6.63.1 StartStyle

Property (get, set): `AnnotationLineEnding StartStyle`
Error code: `IllegalState`

The starting point's style.

Error Code:

`IllegalState` The `PolyLineAnnotation` object or the owning document has already been closed.

6.63.2 EndStyle

Property (get, set): `AnnotationLineEnding EndStyle`
Error code: `IllegalState`

The ending point's style.

Error Code:

`IllegalState` The `PolyLineAnnotation` object or the owning document has already been closed.

6.63.3 LineEndingFill

Property (get, set):	`Paint LineEndingFill`
Error code:	`IllegalState`

The line ending filling paint, applying to both the starting and the ending point.

Error Code:

`IllegalState` The `PolyLineAnnotation` object or the owning document has already been closed.

6.64 PolygonAnnotation Interface

An annotation for drawing polygons.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ DrawingAnnotation
         └─ PolyDrawingAnnotation
             └─ PolygonAnnotation
```

6.64.1 Fill

Property (get, set):	`Paint Fill`
Error code:	`IllegalState`

The filling paint.

Error Code:

`IllegalState` The `PolygonAnnotation` object or the owning document has already been closed.

6.65 PushButtonField Interface

This interface represents a push button form field.

Hierarchy:

```
FormFieldNode
 └─ FormField
     └─ PushButtonField
```

6.66 RadioButton Interface

This interface represents a radio button in a RadioButtonField.

6.66.1 ExportName

Property (get): String ExportName
Error code (get): IllegalState

Specifies the button's name when exporting.

Error Code:

IllegalState The RadioButton or the owning document has already been closed.

6.66.2 Widgets

Property (get): WidgetList Widgets
Error code (get): IllegalState

The button's list of Widget annotations.

Error Code:

IllegalState The RadioButton or the owning document has already been closed.

6.66.3 AddNewWidget

Method: Field AddNewWidget(Rectangle rectangle)
Error codes: IllegalState, UnsupportedOperation, IllegalArgument
License feature: Creation

This method creates a widget (visual manifestation) for the this radio button. The widget is automatically added to this button's `Widgets` and to the radio button form field's `Widgets`, but not to any page.

Parameter:

`rectangle` `[Rectangle]` The widget's target rectangle on the page.

Returns:

The newly created form field widget.

Error Codes:

`IllegalState` The document has already been closed.

`UnsupportedOperation`

- The document is read-only.
- The document contains form fields that have been implicitly copied by `CopyPage` with activated `CopyOption` `CopyFormFields`.

`IllegalArgument` The `rectangle` argument is `null`.

6.67 `RadioButtonField` Interface

This interface represents a radio button form field.

Hierarchy:

```
FormFieldNode
 └─ FormField
     └─ RadioButtonField
```

6.67.1 `Buttons`

Property (get): `RadioButtonlist` `Buttons`
Error code (get): `IllegalState`

This radio button field's `RadioButton`s.

Error Code:

`IllegalState` The `RadioButtonField` or the owning document has already been closed.

6.67.2 CanToggleOff

Property (get, set): `bool CanToggleOff`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

When `true`, then the radio button field's user interface allows a state in which no radio button is selected.

Even if this property is `true`, the `ChosenButton` can be `null`.

Error Codes:

`IllegalState` The `RadioButtonField` or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.67.3 ChosenButton

Property (get, set): `RadioButton ChosenButton`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`, `IllegalArgument`, `IllegalState`

This field's selected `RadioButton`.

Error Codes:

`IllegalState`

- The `RadioButtonField` is `ReadOnly`.
- The `RadioButton` is `null` but the `CanToggleOff` property is `false`.

`UnsupportedOperation` The document associated with the `RadioButtonField` is read-only.

`IllegalArgument` The `RadioButton` does not belong to this radio button field.

6.67.4 AddNewButton

Method: `RadioButton AddNewButton(String exportName)`
Error codes: `IllegalState`, `UnsupportedOperation`

This method creates a new radio button. The created button is automatically added to this radio button field's `Buttons`.

Parameter:

`exportName` `[String]` The radio button's export name.

Returns:

The newly created `RadioButton`.

Error Codes:

`IllegalState` The `RadioButtonField` or the containing document has already been closed.

`UnsupportedOperation`

- The document is read-only.
- The document contains form fields that have been implicitly copied by `CopyPage` with activated `CopyOption` `CopyFormFields`.

6.68 `RadioButtonList` Interface

See Programming Interfaces for more information on how lists are modeled and used in the different interfaces the 3-Heights® PDF Toolbox API provides.

This list contains a `RadioButtonField`'s `RadioButton`s.

6.69 `ShadingElement` Interface

A `ContentExtractor` yields such an element when encountering a PDF shading.

Hierarchy:

```
ContentElement
└─ ShadingElement
```

6.70 `ShapeDrawingAnnotation` Interface

An annotation for drawing basic shapes.

Hierarchy:

```
Annotation
└─ MarkupAnnotation
    └─ DrawingAnnotation
        └─ ShapeDrawingAnnotation
            ├─ CircleAnnotation
            └─ SquareAnnotation
```

6.70.1 Fill

Property (get, set): `Paint Fill`
Error code: `IllegalState`

The shape filling paint.

Error Code:

`IllegalState` The `ShapeDrawingAnnotation` object or the owning document has already been closed.

6.71 SignatureField Interface

The existence of signature fields does not necessarily mean that the document is signed. This depends on the property `IsSigned`.

All properties are taken from information in the PDF, not from the actual digital signature. For some properties, those values are redundant and the value from the actual digital signature should take precedence.

The existence of a signed signature field does not imply, that the signature is valid. The signature is actually not validated at all.

6.71.1 IsSigned

Property (get): `bool IsSigned`
Error code (get): `IllegalState`

Describes whether the field is already signed.

- If `true` the signature field contains an actual digital signature.
- If `false` the signature field is a placeholder that is yet to be signed. In that case, all other properties except `IsVisible` can be ignored.

> **Note:** The digital signature itself is not validated.

Error Code:

`IllegalState` The `SignatureField` object or the owning document has already been closed.

6.71.2 IsVisible

Property (get): `bool IsVisible`
Error code (get): `IllegalState`

If `true`, the signature field has a visual appearance on the page.

Error Code:

`IllegalState` The `SignatureField` object or the owning document has already been closed.

6.71.3 Name

Property (get): `String Name`
Error code (get): `IllegalState`

The name of the person or authority signing the document.

> **Note:** This property should be used with care, as the value might differ from the name as stored in the signer certificate. If that's the case, the name in the signer certificate should be used.

Error Code:

`IllegalState` The `SignatureField` object or the owning document has already been closed.

6.71.4 Location

Property (get): `String Location`
Error code (get): `IllegalState`

The CPU host name or physical location of the signing.

Error Code:

`IllegalState` The `SignatureField` object or the owning document has already been closed.

6.71.5 Reason

Property (get): `String Reason`
Error code (get): `IllegalState`

The reason for the signing.

Error Code:

`IllegalState` The `SignatureField` object or the owning document has already been closed.

6.71.6 ContactInfo

Property (get): `String ContactInfo`
Error code (get): `IllegalState`

Information provided by the signer to enable a recipient to contact the signer to verify the signature; for example, a phone number.

Error Code:

`IllegalState` The `SignatureField` object or the owning document has already been closed.

6.71.7 Date

Property (get): `Date Date`
Error code (get): `IllegalState`

The date and time of signing

Depending on the signature handler, this may be a normal unverified computer time or a time generated in a verifiable way from a secure time server.

> **Note:** This property should be used with care, as the value might differ from the date as stored in the cryptographic signature itself. If that's the case, the name in the signer certificate should be used.

Error Code:

`IllegalState` The `SignatureField` object or the owning document has already been closed.

6.72 SignatureFieldList Interface

See chapter 5 for more information on how lists are modeled.

6.73 SquareAnnotation Interface

A square or rectangle drawing annotation.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ DrawingAnnotation
         └─ ShapeDrawingAnnotation
             └─ SquareAnnotation
```

6.74 SquigglyAnnotation Interface

A squiggly text underlining annotation.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ TextMarkupAnnotation
         └─ SquigglyAnnotation
```

6.75 StampAnnotation Interface

A stamp annotation.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ StampAnnotation
         ├─ CustomStampAnnotation
         └─ TextStampAnnotation
```

6.75.1 Popup

Property (get, set): `AnnotationPopup` `Popup`
Error code: `IllegalState`

The annotation's popup.

Error Code:

`IllegalState` The `StampAnnotation` object or the owning document has already been closed.

6.75.2 PopupPaint

Property (get): `Paint PopupPaint`
Error codes (get): `IllegalState`, `Corrupt`

The paint for the popup.

Error Codes:

`IllegalState` The `StampAnnotation` object or the owning document has already been closed.

`Corrupt` The PDF is corrupt and the annotation's paint cannot be read.

6.76 StickyNoteAnnotation Interface

A sticky note annotation.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ StickyNoteAnnotation
```

6.76.1 Paint

Property (get): `Paint Paint`
Error codes (get): `IllegalState`, `Corrupt`

The paint for the icon an the popup.

Error Codes:

`IllegalState` The `StickyNoteAnnotation` object or the owning document has already been closed.

`Corrupt` The PDF is corrupt and the annotation's paint cannot be read.

6.76.2 Popup

Property (get, set): `AnnotationPopup` `Popup`
Error code: `IllegalState`

The annotation's popup.

Error Code:

`IllegalState` The `StickyNoteAnnotation` object or the owning document has already been closed.

6.77 StrikeThroughAnnotation Interface

An annotation for striking through text.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ TextMarkupAnnotation
         └─ StrikeThroughAnnotation
```

6.78 StringMap Interface

See chapter 5 for more information on how maps (dictionaries) are modeled.

6.79 SubForm Interface

A sub form is a node in the tree of form fields that can have subordinate child fields.

Hierarchy:

```
FormFieldNode
 └─ SubForm
```

6.79.1 Children

Property (get): `FormFieldNodeMap` `Children`
Error code (get): `IllegalState`

The subordinate child form fields of this sub form.

Adding to this map can result in the following errors:

`IllegalState` if the list has already been closed

`UnsupportedOperation` if the document is read-only

`IllegalArgument`

- if the given `FormFieldNode` is `null`.
- if the given `FormFieldNode` has already been closed.
- if the given `FormFieldNode` does not belong to the same document as the map.
- if the given `FormFieldNode` has already been added to a form field node map.
- if the given key is `null`.
- if the given key contains a full stop ".".
- if the given key is already used in this map.

Error Code:

`IllegalState` The `SubForm` or the owning document has already been closed.

6.80 Text Interface

When created with a `Document`'s `CreateText` method, the `Text` object can be modified with a `TextGenerator` and then be used as argument for `PaintText` method of a `ContentGenerator`. In this case, an attempt to extract `TextFragment`'s from the `Text`'s list interface will result in an `IllegalState` error.

When returned as part of a `TextElement` by a `ContentExtractor`, the `Text`'s list interface can be used to extract `TextFragment`s from the text.

See Programming Interfaces for more information on how lists are modeled and used in the different interfaces the 3-Heights® PDF Toolbox API provides.

6.81 TextElement Interface

A `ContentExtractor` yields such an element when encountering a PDF text.

Hierarchy:

```
ContentElement
└─ TextElement
```

6.81.1 Text

Property (get): `Text Text`
Error code (get): `IllegalState`

Use this property to access this text element's text.

Error Code:

`IllegalState` The `TextElement` or the owning document has already been closed.

6.82 TextField Interface

Hierarchy:

```
FormFieldNode
 └─ FormField
     └─ TextField
         ├─ CombTextField
         └─ GeneralTextField
```

6.82.1 Alignment

Property (get, set): `TextAlignment Alignment`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

The alignment of the contained text with respect to the bounding `Rectangle`.

Error Codes:

`IllegalState`

- (get, set) The `TextField` or the owning document has already been closed.
- (set) The `TextField` has `Widgets`.

`UnsupportedOperation` The document is read-only.

6.82.2 FontSize

Property (get, set): `double? FontSize`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`, `IllegalArgument`

The font size used in this text field. If `null`, then the font size is choosen automatically by the PDF viewer.

Error Codes:

`IllegalState`

- (get, set) The `TextField` or the owning document has already been closed.
- (set) The `TextField` has `Widgets`.

`UnsupportedOperation` The document is read-only.

`IllegalArgument` The given value is smaller than `0.0`.

6.82.3 MaxLength

Property (get, set): `int MaxLength`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`, `IllegalArgument`

The maximal length of the `Text`. `-1` signifies that the maximal length is not defined.

When setting this property, the this field's text is truncated to the given length.

Error Codes:

`IllegalState`

- (get, set) The `TextField` or the owning document has already been closed.
- (set) The `TextField` has `Widgets`.

`UnsupportedOperation` The document is read-only.

`IllegalArgument`

- If this is `GeneralTextField` and the given value is smaller than `-1`.
- If this is a `CombTextField` and the given value is smaller than `0`.

6.82.4 Text

Property (get, set): `String Text`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

This field's text.

Error Codes:

`IllegalState`

- (get, set) The `TextField` or the owning document has already been closed.
- (set) The `TextField` is `ReadOnly`.
- (set) The `TextField` has `Widgets`.

`UnsupportedOperation` The document associated with the `TextField` is read-only.

6.83 TextFragment Interface

6.83.1 BoundingBox

Property (get): `Rectangle BoundingBox`
Error code (get): `IllegalState`

This is the rectangle in untransformed coordinates that encompasses all the parts of the text fragment. Use the `Transformation` to compute the transformed (actual) coordinates with respect to the parent page or group content.

Error Code:

`IllegalState` The `TextFragment` object or the owning `Text` object has already been closed.

6.83.2 Fill

Property (get): `FillParams Fill`
Error code (get): `IllegalState`

This specifies the text fragment's parameters for filling the text or `null` if the text is not filled.

Error Code:

`IllegalState` The `TextFragment` object or the owning `Text` object has already been closed.

6.83.3 Stroke

Property (get): `StrokeParams Stroke`
Error code (get): `IllegalState`

This specifies the text fragment's parameters for stroking the text or `null` if the text is not stroked.

Error Code:

`IllegalState` The `TextFragment` object or the owning `Text` object has already been closed.

6.83.4 Transformation

Property (get): `Transformation Transformation`
Error code (get): `IllegalState`

This is the coordinate transformation applicable to this text fragment. Use this transform to compute the actual location of the text fragment's bounding box on the containing page or group.

Error Code:

`IllegalState` The `TextFragment` object or the owning `Text` object has already been closed.

6.83.5 UnicodeString

Property (get): `String UnicodeString`
Error code (get): `IllegalState`

This is the extracted Unicode string. It may contain Unicode replacement characters if the text is not extractable.

Error Code:

`IllegalState` The `TextFragment` object or the owning `Text` object has already been closed.

6.84 TextGenerator Interface

6.84.1 TextGenerator Constructor

Method: `TextGenerator TextGenerator(Text text, Font font, double fontSize, Point position)`
Error code: `IllegalArgument`
License feature: `Creation`

Create a new text generator for appending text to a text content object.

All parameters that cannot be set in the constructor are set to default values:

- Rendering: fill text with black paint
- CharSpacing: `0`
- WordSpacing: `0`
- HorizontalScaling: `1`
- Leading: 1.2 times the `FontSize`
- Rise: `0`

- Stroke: `null`
- IntersectClipping: `false`

Error Code:

`IllegalArgument`

- The `text` argument is `null`.
- The `text` argument has already been closed.
- The `text` and `font` arguments are associated with different documents.
- The `font` argument is `null`.
- The `font` argument has already been closed.
- The owning document of the `font` argument has already been closed.

6.84.2 CharSpacing

Property (set): `double CharSpacing`
Default: `0`
Error code (set): `IllegalState`

Set the current character spacing.

When the glyph for each character in the string is rendered, the character spacing is added to the horizontal or vertical component of the glyph's displacement, depending on the writing mode. It is subject to scaling by the horizontal scaling if the writing mode is horizontal.

Error Code:

`IllegalState` The `TextGenerator` object or the owning `Text` object has already been closed.

6.84.3 Close

Method: `void Close()`
Error code: `IllegalState`

Close the Generator.

The generated object is not finalized until this method is called.

All native data structures that belong to the generator are freed.

After closing the generator, any call to a method of the generator will result in an `IllegalState` error.

Error Code:

`IllegalState` The generator has already been closed.

6.84.4 Font

Property (set): `Font Font`
Error codes (set): `IllegalState`, `IllegalArgument`

Set the current font.

The font is used for all subsequent `Show` and `ShowLine` calls.

Error Codes:

`IllegalState` The `TextGenerator` object or the owning `Text` object has already been closed.

`IllegalArgument`

- The argument is `null`.
- The argument has already been closed.
- The argument is associated with a different document.

6.84.5 FontSize

Property (set): `double FontSize`
Error code (set): `IllegalState`

Set the current font size.

The font size is used for all subsequent `Show` and `ShowLine` calls.

Note that this sets the font size only. Also use `Leading` to set the leading.

Error Code:

`IllegalState` The `TextGenerator` object or the owning `Text` object has already been closed.

6.84.6 GetWidth

Method: `double GetWidth(String text)`
Error codes: `IllegalState`, `IllegalArgument`

Get the width of a text string.

The width of a text string as if it would be shown with the current settings.

Returns:

The width of the text.

Parameter:

`text` `[String]` The text fragment.

Error Codes:

`IllegalState` The `TextGenerator` object or the owning `Text` object has already been closed.

`IllegalArgument` The `text` argument is `null`.

6.84.7 HorizontalScaling

Property (set): `double HorizontalScaling`
Default: `1`
Error code (set): `IllegalState`

Set the current horizontal scaling.

The horizontal scaling parameter adjusts the width of glyphs by stretching or compressing them in the horizontal direction. Its value is specified relative to the normal width of the glyphs, with 1 being the normal width.

Error Code:

`IllegalState` The `TextGenerator` object or the owning `Text` object has already been closed.

6.84.8 Leading

Property (set): `double Leading`
Default: `1.2 times the initial font size`
Error code (set): `IllegalState`

Set the current leading.

The leading parameter specifies the vertical distance between the baselines of adjacent lines of text. It affects only the method `ShowLine`.

Error Code:

`IllegalState` The `TextGenerator` object or the owning `Text` object has already been closed.

6.84.9 MoveTo

Method: `void MoveTo(Point position)`
Error codes: `IllegalState`, `IllegalArgument`

Move the current position.

This also also sets the beginning of the current line to the specified position, which will affect the `ShowLine` method.

Parameter:

`position` `[Point]` The target position.

Error Codes:

`IllegalState` The `TextGenerator` object or the owning `Text` object has already been closed.

`IllegalArgument` The `position` argument is `null`.

6.84.10 Rise

Property (get, set): `double Rise`
Default: `0`
Error code: `IllegalState`

Set the current rise of the baseline.

Text rise specifies the distance to move the baseline up or down from its default location. Positive values of text rise move the baseline up. Adjustments to the baseline are useful for drawing superscripts or subscripts.

Error Code:

`IllegalState` The `TextGenerator` object or the owning `Text` object has already been closed.

6.84.11 SetRendering

Method: `void SetRendering(Paint fill, StrokeParams stroke, bool intersectClipping)`
Error codes: `IllegalState`, `IllegalArgument`, `UnsupportedOperation`

Set text rendering parameters.

The text is first filled and then stroked, then the clip path is intersected.

The default rendering of text is filled with black paint.

> **Note:** The blend modes for filling and stroking must be the same.

Parameters:

`fill` `[Paint]` The fill paint or `null` if the text should not be filled.

For PDF/A documents it is recommended to set an output intent and use a fill with either a device gray color space or one that matches the output intent's.

stroke [StrokeParams] The stroke properties or null if the text should not be stroked.

intersectClipping [bool] If true the current clip path is intersected.

Error Codes:

IllegalState The TextGenerator object or the owning Text object has already been closed.

IllegalArgument

- The fill argument has already been closed.
- The fill argument belongs to a different document.
- The field Paint of argument stroke is null.
- The field Paint of argument stroke has already been closed.
- The field Paint of argument stroke belongs to a different document.

UnsupportedOperation The Paint objects associated with the fill and stroke arguments use different blend modes.

6.84.12 Show

Method: void Show(String text)
Error codes: IllegalState, IllegalArgument

Show a text string.

The text is shown using the current settings.

Parameter:

text [String] The text to be shown.

Error Codes:

IllegalState The TextGenerator object or the owning Text object has already been closed.

IllegalArgument The text argument is null.

6.84.13 ShowLine

Method: void ShowLine(String text)
Error codes: IllegalState, IllegalArgument

Show a text string and go to the next line.

Parameter:

`text` `[String]` The text to be shown.

Error Codes:

`IllegalState` The `TextGenerator` object or the owning `Text` object has already been closed.

`IllegalArgument` The `text` argument is `null`.

6.84.14 WordSpacing

Property (get, set): `double WordSpacing`
Default: `0`
Error code: `IllegalState`

Set the current word spacing.

Word spacing works the same way as character spacing, but applies only to the space character, code 32.

Error Code:

`IllegalState` The `TextGenerator` object or the owning `Text` object has already been closed.

6.85 TextInsertAnnotation Interface

An annotation that marks a point to insert text.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ TextInsertAnnotation
```

6.85.1 Paint

Property (get): `Paint Paint`
Error codes (get): `IllegalState`, `Corrupt`

The paint for the annotation and the popup.

Error Codes:

`IllegalState` The `TextInsertAnnotation` object or the owning document has already been closed.

`Corrupt` The PDF is corrupt and the annotation's paint cannot be read.

6.85.2 Popup

Property (get, set): `AnnotationPopup` `Popup`
Error code: `IllegalState`

The annotation's popup.

Error Code:

`IllegalState` The `TextInsertAnnotation` object or the owning document has already been closed.

6.86 `TextMarkupAnnotation` Interface

An annotation that marks up part(s) of a text.

Hierarchy:

```
Annotation
└─ MarkupAnnotation
    └─ TextMarkupAnnotation
        ├─ HighlightAnnotation
        ├─ SquigglyAnnotation
        ├─ StrikeThroughAnnotation
        └─ UnderlineAnnotation
```

6.86.1 `Paint`

Property (get): `Paint` `Paint`
Error codes (get): `IllegalState`, `Corrupt`

The paint for the annotation and the popup.

Error Codes:

`IllegalState` The `TextMarkupAnnotation` object or the owning document has already been closed.

`Corrupt` The PDF is corrupt and the annotation's paint cannot be read.

6.86.2 Popup

Property (get, set): `AnnotationPopup` `Popup`
Error code: `IllegalState`

The annotation's popup.

Error Code:

`IllegalState` The `TextMarkupAnnotation` object or the owning document has already been closed.

6.87 TextStampAnnotation Interface

A text stamp annotation.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ StampAnnotation
         └─ TextStampAnnotation
```

6.87.1 TextType

Property (get, set): `TextStampType` `TextType`
Error code: `IllegalState`

This property defines the predefined text for this text stamp.

Error Code:

`IllegalState` The `TextStampAnnotation` object or the owning document has already been closed.

6.88 Transformation Interface

The transformation matrix in PDF is specified by six numbers. All information about orientation, rotation, scaling, skewing and translation can be calculated based on these six numbers. The 3-Heights® PDF Toolbox API provides methods which compute these values.

The actual matrix is $M = \begin{bmatrix} m11 & m12 & 0 \\ m21 & m22 & 0 \\ dx & dy & 1 \end{bmatrix}$

This matrix is used to define a transformation of a vector $[\,x\ y\ 1\,]$ to a vector $\left[\,x'\ y'\ 1\,\right] = [\,x\ y\ 1\,] \cdot M$, where (x, y) is the original point and $(x',\ y')$ is the transformed point on the page.

6.88.1 `Transformation` Constructor

Method: `Transformation()`

Method: `Transformation(double m11, double m12, double m21, double m22, double dx, double dy)`

Method: `Transformation(Transformation other)`

Create an new transformation.

Without any arguments, the identity transform is created.

6.88.2 `Concatenate`

Method: `void Concatenate(Transformation transformation)`
Error code: `IllegalState`

Concatenate transformation with other transformation.

Parameter:

`transformation` [`Transformation`] The other transformation.

Error Code:

`IllegalState` The `Transformation` object has already been closed.

6.88.3 `Invert`

Method: `void Invert()`
Error code: `IllegalState`

A transform usually maps from the transformed coordinate system to the untransformed coordinate system. Use this method to create the reverse transform.

Error Code:

`IllegalState`

- The `Transformation` object has already been closed.
- The transform is not invertible.

6.88.4 Rotate

Method: `void Rotate(double a)`
Error code: `IllegalState`

Rotate.

Rotations are produced by `[cos(a) sin(a) -sin(a) cos(a) 0 0]`, which has the effect of rotating the coordinate system axes by an angle a counterclockwise.

Parameter:

a `[double]` The angle in degrees.

Error Code:

`IllegalState` The `Transformation` object has already been closed.

6.88.5 RotateAround

Method: `void RotateAround(double angle, Point center)`
Error code: `IllegalState`

This method is equivalent to the following sequence:

- `Translate` to `center`.
- `Rotate` by `angle`.
- `Translate` "back" to `(-center.x, -center.y)`.

Parameters:

angle `[double]` The angle in degrees.

center `[Point]` The center in the current coordinate system.

Error Code:

`IllegalState` The `Transformation` object has already been closed.

6.88.6 Scale

Method: `void Scale(double sx, double sy)`
Error code: `IllegalState`

Scale.

Scaling is obtained by `[sx 0 0 sy 0 0]`. This scales the coordinates so that 1 unit in the horizontal and vertical dimensions of the new coordinate system is the same size as `sx` and `sy` units, respectively, in the previous coordinate system.

Parameters:

`sx` `[double]` Horizontal scaling.

`sy` `[double]` Vertical scaling.

Error Code:

`IllegalState` The `Transformation` object has already been closed.

6.88.7 Skew

Method: `void Skew(double alpha, double beta)`
Error code: `IllegalState`

Skew.

Skew is specified by `[1 tan(α) tan(β) 1 0 0]`, which skews the x axis by an angle α and the y axis by an angle β.

Parameters:

`alpha` `[double]` Angle α in degrees.

`beta` `[double]` Angle β in degrees.

Error Code:

`IllegalState` The `Transformation` object has already been closed.

6.88.8 TransformPoint

Method: `Point TransformPoint(Point original)`

Apply transformation to point.

Use this method to compute the transformed point from the original point.

6.88.9 Translate

Method: `void Translate(double tx, double ty)`
Error code: `IllegalState`

Translate.

Translations are specified as `[1 0 0 1 tx ty]`, where `tx` and `ty` are the distances to translate the origin of the coordinate system in the horizontal and vertical dimensions, respectively.

Parameters:

`tx` `[double]` Horizontal translation.

`ty` `[double]` Vertical translation.

Error Code:

`IllegalState` The `Transformation` object has already been closed.

6.89 UnderlineAnnotation Interface

A text underlining annotation.

Hierarchy:

```
Annotation
 └─ MarkupAnnotation
     └─ TextMarkupAnnotation
         └─ UnderlineAnnotation
```

6.90 ViewerSettings Interface

6.90.1 PageDisplay

Property (get, set): `PageDisplay PageDisplay`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

The positional arrangment for displaying pages when opening the document in a viewer.

If `null` then the viewer acts according to it's default behavior.

Error Codes:

`IllegalState` The `ViewerSettings` object or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.90.2 ActivePane

Property (get, set): `ViewerNavigationPane? ActivePane`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

The initially visible side pane when opening the document in a viewer.

If `null` then the viewer acts according to it's default behavior.

Error Codes:

`IllegalState` The `ViewerSettings` object or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.90.3 FullScreen

Property (get, set): `bool FullScreen`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

If `true` then opening the document will make the viewer try to enter full screen mode.

Error Codes:

`IllegalState` The `ViewerSettings` object or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.90.4 HideToolbar

Property (get, set): `bool HideToolbar`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

If `true` then opening the document will instruct the viewer to hide its toolbar.

Error Codes:

`IllegalState` The `ViewerSettings` object or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.90.5 Menubar

Property (get, set): `bool Menubar`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

If `true` then opening the document will instruct the viewer to hide its menubar.

Error Codes:

`IllegalState` The `ViewerSettings` object or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.90.6 DisplayDocumentTitle

Property (get, set): `bool DisplayDocumentTitle`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

If `true` then opening the document will instruct the viewer to display the document's title from the metadata instead of its file name.

Error Codes:

`IllegalState` The `ViewerSettings` object or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.91 WebLink Interface

A link annotation for external links.

Hierarchy:

```
Annotation
└─ Link
    └─ WebLink
```

6.91.1 WebLink Constructor

Method: `WebLink(Rectangle rectangle)`
Error code: `IllegalArgument`
License feature: `Annotation`

Create a new external link that has no `Uri` yet.

Parameter:

`rectangle` `[Rectangle]` The location on the page.

Error Code:

`IllegalArgument` The `rectangle` argument is `null`.

6.91.2 Uri

Property (get, set): `String Uri`
Error code (get): `IllegalState`
Error codes (set): `IllegalState`, `UnsupportedOperation`

The link's target URI (uniform resource identifier).

Error Codes:

`IllegalState` The `InternalLink` object or the owning document has already been closed.

`UnsupportedOperation` The document is read-only.

6.92 Widget Interface

Widget annotations are the visual manifestation of `FormField`s on the document's pages.

Hierarchy:

```
Annotation
└─ Widget
```

6.93 WidgetList Interface

See Programming Interfaces for more information on how lists are modeled and used in the different interfaces the 3-Heights® PDF Toolbox API provides.

This list contains `Widget` annotations.

6.94 Structures

6.94.1 EncryptionParams Struct

OwnerPassword [string] The owner password of the document.

This password is also referred to as the author's password and grants full access to the document. Not only can the document be opened and read, it also allows for changing the document's security settings (access permission and passwords).

UserPassword [string] The user password of the document.

protects the document against unauthorized opening and reading. If a PDF document is protected by a user password, either the user or owner password must be provided to open and read the document. If a document has a user password, it must have an owner password as well. If no owner password is defined, the owner password is the same as the user password.

UserPermissions [Permission] The user permissions.

What operations in a PDF document are granted is controlled via these permission flags. In order to set permission flags, the PDF document must be encrypted and have an owner password. The owner password is required to initially set or later change the permission flags.

6.94.2 FillParams Struct

Paint [Paint] The filling paint. (Default `null`)

The paint defines the visual style to be used for filling a shape.

See `Paint`.

FillRule [InsideRule] The inside rule. (Default `NonzeroWindingNumber`)

The inside rule defines which parts of a shape path used for filling.

See `InsideRule`.

6.94.3 PageDisplay Struct

PageLayout [PageLayout] The page layout.

Continuous [bool] Whether pages are shown in single-page mode or in a continuous column.

6.94.4 Point Struct

X [double] The X value (horizontal coordinate).

Y [double] The Y value (vertical coordinate).

6.94.5 Rectangle Struct

Left [double] The Left value.

Bottom [double] The Bottom value.

Right [double] The Right value.

Top [double] The Top value.

6.94.6 Size Struct

Width [double] The Width value.

Height [double] The Height value.

6.94.7 StrokeParams Struct

DashArray [double[]] The dash array of a line dash pattern.

The line dash pattern controls the pattern of dashes and gaps used to stroke paths.

DashPhase [double] The dash phase of a line dash pattern.

The dash phase is the offset of the pattern and can be larger as the pattern itself.

LineCapStyle [LineCapStyle] The line cap style.

The line cap style specifies the shape to be used at the end of open sub-paths and dashes when they are stroked.

See LineCapStyle

LineJoinStyle [LineJoinStyle] The line join style.

The line join style specifies the shape to be used at the corners of paths that are stroked.

See LineJoinStyle

LineWidth [double] The line width in user units.

MiterLimit [double] The miter limit.

The miter limit imposes a maximum on the ratio of the miter length to the line width, which can be fairly large when two line segments meet at a sharp angle. When the limit is exceeded, the line join is converted from a miter to a bevel.

Paint [Paint] The paint used to color the line.

See Paint

6.94.8 TransparencyParams Struct

BlendMode [BlendBode] The blend mode. (Default Normal)

The blend mode defines the computation rule for compositing content with its backdrop.

See BlendMode.

ConstAlpha [double] The constant alpha value. (Default 1.0)

This value can range from 0.0 (fully transparent) to 1.0 (fully opaque).

6.95 Enumerations

6.95.1 `AnnotationLineEnding` Enumeration

Specifies the type of line termination for `LineAnnotation`s and `PolyLineAnnotation`s.

`None`

`OpenArrow`

`ClosedArrow`

`Square`

`Circle`

`Diamond`

`Butt`

`ReversedOpenArrow`

`ReversedClosedArrow`

`Slash`

6.95.2 `BlendMode` Enumeration

`Normal` Selects the source color, ignoring the backdrop.

`Multiply` Multiplies the backdrop and source color values.

The result color is always at least as dark as either of the two constituent colors.

When working with additive colors, multiplying any color with black produces black while multiplying with white leaves the original color unchanged.

For subtractive colors, the maximum tint value used for all colorants of the color space acts as black does for additive spaces.

Painting successive overlapping objects with a color other than black or white produces progressively darker colors.

`Screen` Multiplies the complements of the backdrop and source color values, then complements the result.

The result color is always at least as light as either of the two constituent colors.

When working with additive colors, screening any color with white produces white while screening with black leaves the original color unchanged.

For subtractive colors, the maximum tint value of all colorants of the color space acts as black does for additive spaces.

The effect is similar to projecting multiple photographic slides simultaneously onto a single screen.

`Darken` Selects the darker of the backdrop and source colors.

The backdrop is replaced with the source where the source is darker; otherwise, it is left unchanged.

`Lighten` Selects the lighter of the backdrop and source colors.

The backdrop is replaced with the source where the source is lighter; otherwise, it is left unchanged.

`ColorDodge` Brightens the backdrop color to reflect the source color.

Painting with black produces no change.

`ColorBurn` Darkens the backdrop color to reflect the source color.

Painting with white produces no change.

`HardLight` Multiplies or screens the colors, depending on the source color value.

The effect is similar to shining a harsh spotlight on the backdrop.

`SoftLight` Darkens or lightens the colors, depending on the source color value.

The effect is similar to shining a diffused spotlight on the backdrop.

`Overlay` Multiplies or screens the colors, depending on the backdrop color value.

Source colors overlay the backdrop while preserving its highlights and shadows. The backdrop color is not replaced but is mixed with the source color to reflect the lightness or darkness of the backdrop.

`Difference` Subtracts the darker of the two constituent colors from the lighter color.

Painting with white inverts the backdrop color; painting with black produces no change.

For subtractive colors, the maximum tint value for all colourants of the color space acts as black does for additive spaces.

This blend mode is not white-preserving.

`Exclusion` Produces an effect similar to that of the `Difference` mode but lower in contrast.

Painting with white inverts the backdrop color; painting with black produces no change.

For subtractive colors, the maximum tint value for all colourants of the color space acts as black does for additive spaces.

`Hue` Creates a color with the hue of the source color and the saturation and luminosity of the backdrop color.

This blend mode is not separable.

`Saturation` Creates a color with the saturation of the source color and the hue and luminosity of the backdrop color.

Painting with this mode in an area of the backdrop that is a pure gray (no saturation) produces no change.

This blend mode is not separable.

`Color` Creates a color with the hue and saturation of the source color and the luminosity of the backdrop color.

This preserves the gray levels of the backdrop and is useful for coloring monochrome images or tinting color images.

This blend mode is not separable.

`Luminosity` Creates a color with the luminosity of the source color and the hue and saturation of the backdrop color.

This produces an inverse effect to that of the `Color` mode.

This blend mode is not separable.

6.95.3 ColorSpaceType Enumeration

`DeviceGray`

`DeviceRGB`

`DeviceCMYK`

`CalGray`

`CalRGB`

`Lab`

`ICCBased`

`Indexed`

`Separation`

`DeviceN`

`NChannel`

6.95.4 Conformance Enumeration

`Unknown` The conformance is unkown or automatically determined.

`Pdf10` PDF Version 1.0

`Pdf11` PDF Version 1.1

`Pdf12` PDF Version 1.2

`Pdf13` PDF Version 1.3

`Pdf14` PDF Version 1.4 (corresponds to Acrobat 5)

`Pdf15` PDF Version 1.5

`Pdf16` PDF Version 1.6 (corresponds to Acrobat 7)

`Pdf17` PDF Version 1.7, ISO 32000-1

`Pdf20` PDF Version 2.0, ISO 32000-2

`PdfA1B` PDF/A-1b, ISO 19005-1, Level B conformance

`PdfA1A` PDF/A-1a, ISO 19005-1, Level A conformance

`PdfA2B` PDF/A-2b, ISO 19005-2, Level B conformance

`PdfA2U` PDF/A-2u, ISO 19005-2, Level U conformance

`PdfA2A` PDF/A-2a, ISO 19005-2, Level A conformance

`PdfA3B` PDF/A-3b, ISO 19005-3, Level B conformance

`PdfA3U` PDF/A-3u, ISO 19005-3, Level U conformance

`PdfA3A` PDF/A-3a, ISO 19005-3, Level A conformance

6.95.5 CopyOption Enumeration

`CopyLinks` Copy links (document internal and external links).

`CopyAnnotations` Copy interactive annotations such as sticky notes or highlight annotations. This does not include link annotations and form field widgets.

`CopyFormFields` Copy interactive form fields.

Note that when merging multiple documents with form fields, it is important that no two different form fields have the same name. Otherwise one of the fields must be renamed. Consider using `FlattenFormFields` when merging multiple forms.

`CopyOutlines` Copy all outline items (bookmarks) that point to the copied page.

The structure of the outline tree in the output document will be the same as in the input document, regardless of the order in which pages are copied.

`CopyLogicalStructure` Copy logical structure information.

Logical structure information in a PDF defines the structure of content, such as titles, paragraphs, figures, reading order, tables or articles. Logical structure elements can be "tagged" with descriptions or alternative text. E.g. “tagging” allows the contents of an image to be described to the visually impaired.

It is recommended to use this option, if all input documents are “tagged”. Otherwise this could be deactivated in order to create smaller output files and get a much better performance. This option is required for PDF/A level A conformance (e.g. PDF/A-1a, PDF/A-2a, PDF/A-3a).

`CopyNamedDestinations` Copy named destinations.

A document may contain a mapping of names to destinations within the document. These names can then be used in link annotations or outlines in order to refer to destinations within the document.

Links within the document will work regardless of the state of this flag. If `CopyNamedDestinations` is not used, all named destinations of the input document are removed and all internal named destinations converted to regular destinations. This is much faster than copying named destinations.

If a document is split into multiple documents with the intention of merging the pieces back together at a later time, this flag should be used. If the document uses named destinations, links between the pieces will work after merging if `CopyNamedDestinations` is used.

`CopyAssociatedFiles` Copy associated files.

In PDF/A-3, associated files are embedded files that are associated to a PDF object such as a page or the document. When copying pages with this flag enabled, then any embedded files associated with the page range are also copied.

`FlattenAnnotations` Flatten annotations preserves the visual appearance of annotations, but discards all interactive elements.

`FlattenFormFields` Flatten form fields preserves the visual appearance of form fields, but discards all interactive elements.

`FlattenSignatureAppearances` Flatten the visual appearance of signed signature fields.

A digital signature consists of two parts: First, a cryptographic part that includes a hash value based on the content of the document that is being signed. If the document is modified at a later time, the computed hash value is no longer correct and the signature becomes invalid, i.e. the validation will fail and will report that the document has been modified since the signature has been applied. Second, an optional visual appearance on a page of the PDF document. The signature appearance can be useful to indicate the presence of a digital signature by a particular signer.

Processing the PDF with 3-Heights® PDF Toolbox API breaks the signature, and therefore the cryptographic part needs to be removed. In general, the visual appearance is regarded as worthless without the cryptographic part, it is removed by default. The visual appearance can be preserved by setting the flag `FlattenSignatureAppearances`.

`OptimizeResources` Find and merge redundant resources from different input files. Equal fonts, images and color spaces are detected. By activating this feature, much smaller output files are created, if similar files are merged. However, the merging process uses more time and memory resources.

`MergeOCGs` Merge optional content groups.

In PDF, optional content groups (also known as “layers”) are identified by a name. When merging PDFs with this flag enabled, then identically named OCGs in different input documents are collapsed.

`SeparateAcroForms` Separate Acroform form field identifiers.

When merging PDFs that contain Acroform form fields any of which have the same identifier, then the fields are regarded as distinct and new identifiers are generated in the output document.

6.95.6 DeviceColorSpaceType Enumeration

Gray One channel gray.

RGB Three channels red, green, blue.

CMYK Four channels cyan, magenta, yellow, key (black).

6.95.7 ErrorCode Enumeration

See Programming Interfaces for more information about how these codes are mapped to exceptions in the .NET and java interface.

Logic errors

These codes denote errors in the application program logic and should never happen at runtime.

UnsupportedOperation The requested method or property is not supported.

IllegalState The object is in a state, where the requested object or property cannot be called.

IllegalArgument The method was called using an illegal argument.

Environmental errors

Generic The error is not further specified.

Fatal A fatal error occurred.

License Licensing error.

NotFound The requested item or resource could not be found.

IO Error while reading or writing from a stream.

UnknownFormat The format is unknown.

Corrupt The data is corrupt.

Password The resource or document is protected by a password.

Conformance A conformance mismatch happened.

UnsupportedFeature The file contains an unsupported feature.

Infrastructure An infrastructure error occurred.

Processing The file cannot be processed.

Exists The item already exists.

6.95.8 FileAttachmentIcon Enumeration

Specifies the type of icon displayed on a page for a `FileAttachmentAnnotation`.

Graph

PushPin

`Paperclip`

`Tag`

`CustomIcon`

6.95.9 InsideRule Enumeration

`NonzeroWindingNumber` The nonzero winding number rule determines whether a given point is inside a path by conceptually drawing a ray from that point to infinity in any direction and then examining the places where a segment of the path crosses the ray. Starting with a count of 0, the rule adds 1 each time a path segment crosses the ray from left to right and subtracts 1 each time a segment crosses from right to left. After counting all the crossings, if the result is 0 then the point is outside the path; otherwise it is inside.

`EvenOdd` The even-odd rule determines the "insideness" of a point by drawing a ray from that point in any direction and simply counting the number of path segments that cross the ray, regardless of direction. If this number is odd, the point is inside; if even, the point is outside. This yields the same results as the nonzero winding number rule for paths with simple shapes, but produces different results for more complex shapes.

6.95.10 LineCapStyle Enumeration

The line cap style specifies the shape to be used at the end of open sub-paths and dashes when they are stroked.

`Butt` Butt cap

`Round` Round cap

`Square` Projecting square cap

6.95.11 LineJoinStyle Enumeration

The line join style specifies the shape to be used at the corners of paths that are stroked.

`Miter` Miter join

`Round` Round join

`Bevel` Bevel join

6.95.12 PageLayout Enumeration

This enumeration is used to identify the horizontal arrangement for displaying pages.

`OnePage` Displays one page.

`TwoPage` Displays two pages side by side.

`TwoPageFirstPageSingle` Use `OnePage` display for the first page and `TwoPage` display for the remaining pages.

6.95.13 Permission Enumeration

An enumeration for permission flags. If a flag is set, the permission is granted.

`Print` Low resolution printing

`Modify` Changing the document

`Copy` Content copying or extraction

`Annotate` Annotations

`FillForms` Filling of form fields

`SupportDisabilities` Support for disabilities

`Assemble` Document assembly

`DigitalPrint` High resolution printing

6.95.14 `Rotation` Enumeration

`Clockwise` Rotate 90° clockwise.

`CounterClockwise` Rotate 90° counter-clockwise.

`NoRotation` No rotation.

`UpsideDown` Rotate 180° (upside-down).

6.95.15 `TextAlignment` Enumeration

`Left` Flush-left text alignment.

`Center` Centered text alignment.

`Right` Flush-right text alignment.

6.95.16 `TextStampType` Enumeration

Specifies the text displayed for predefined `TextStampAnnotation`s.

`Approved`

`Experimental`

`NotApproved`

`AsIs`

`Expired`

`NotForPublicRelease`

`Confidential`

`Final`

`Sold`

`Departmental`

`ForComment`

`TopSecret`

`Draft`

`ForPublicRelease`

`CustomStampText`

6.95.17 UngroupingSet Enumeration

This enumeration is used to control the behavior of content extraction. Groups in the content can either be extracted as `GroupElement`s, or their content can be un-grouped, in which case groups' content elements are extracted as if not belonging to a group.

`None` Groups are extracted as `GroupElement`s.

`SafelyUngroupable` Un-grouping is restricted to those groups that can be un-grouped without visual loss.

6.95.18 ViewerNavigationPane Enumeration

This enumeration is used to specify an informational side pane in a viewer used for document navigation or displaying document wide information.

`None`

`Outlines`

`Thumbnails`

`Layers`

`EmbeddedFiles`

7 Version History

Some of the documented changes below may be preceded by a marker that specifies the interface technologies the change applies to. E.g. [C, Java] applies to the C and the Java interface.

7.1 Patches in Version 6.12

Patch 6.12.10

- **Fixed** an issue when setting XMP metadata with certain data lengths.
- **Improved** performance reading files with excessive number of compressed object streams.

Patch 6.12.5

- **Fixed** processing of input documents with named destinations that have an empty name.
- **Fixed** processing of input documents that have cyclic references in the document outline.
- **Fixed** an issue where the `update` command of the command line license manager utility (`licmgr`) would connect to the wrong licensing server.

Patch 6.12.2

- **Fixed** an error when filling in form fields with text for which no suitable font is available in the installed fonts.

Patch 6.12.1

- **Improved** Generation of annotation appearances.
- **Improved** support for PDF documents with corrupt and invalidly constructed graphics objects.
- **Improved** methods `CreatePolyLineAnnotation` and `CreatePolygonAnnotation` of the `Document` interface to be less strict on `Path` parameter validation.

7.2 Changes in Version 6

- [.NET] **Improved**`ErrorCodeException` which is now serializable (except for the .NET Standard 1.0 target).
- **Improved** search algorithm for installed fonts: User fonts under Windows are now also taken into account.
- [Java] **Changed** minimal supported Java language version to 7 [previously 6].
- [.NET] **New** availability of this product as NuGet package for Windows, macOS and Linux.
- [.NET] **New** support for .NET Core versions 1.0 and higher. The support is restricted to a subset of the operating systems supported by .NET Core, see Operating Systems.
- [.NET] **Changed** platform support for NuGet packages: The platform "AnyCPU" is now supported for .NET Framework projects.
- [Java] **New** static method `fromValue` for all non-flags enums.
- [.NET, C, Java] **New** interfaces for PDF annotations:

  - `Link`
  - `WebLink`
  - `InternalLink`
  - `MarkupAnnotation`
  - `DrawingAnnotation`
  - `LineAnnotation`
  - `PolyDrawingAnnotation`
  - `FreeDrawingAnnotation`
  - `PolyLineAnnotation`
  - `PolygonAnnotation`
  - `ShapeDrawingAnnotation`
  - `CircleAnnotation`
  - `SquareAnnotation`
  - `FileAttachmentAnnotation`
  - `FreeTextAnnotation`
  - `StickyNoteAnnotation`
  - `TextInsertAnnotation`
  - `TextMarkupAnnotation`
  - `HighlightAnnotation`
  - `SquigglyAnnotation`
  - `StrikeThroughAnnotation`
  - `UnderlineAnnotation`
  - `TextStampAnnotation`
  - `MarkupInfo`
  - `MarkupInfoList`
  - `AnnotationPopup`
- [.NET, C, Java] **New** enums for certain annotations:
  - `FileAttachmentIcon`
  - `TextStampType`
  - `AnnotationLineEnding`
- [.NET, C, Java] **New** interface `ViewerSettings` for viewer settings.
- [.NET, C, Java] **New** enum `PageLayout` for viewer settings.
- [.NET, C, Java] **New** enum `ViewerNavigationPane` for viewer settings.
- [.NET, C, Java] **New** struct `PageDisplay` for viewer settings.
- [.NET] **Deprecated** method `Dispose` for all interfaces except for `Document`, `ContentGenerator`, `PathGenerator` and `TextGenerator`.
- [Java] **Deprecated** method `close` for all interfaces except for `Document`, `ContentGenerator`, `PathGenerator` and `TextGenerator`.
- [.NET, C, Java] **Removed** deprecated interface `Form`.

Interface `Annotation`

- [.NET, C, Java] **New** property `Hidden`.
- [.NET, C, Java] **New** property `DoPrint`.
- [.NET, C, Java] **New** property `NoZoom`.
- [.NET, C, Java] **New** property `NoRotate`.

Interface ContentGenerator

- [Java] **Changed** inheritance. Interface now inherits from `AutoCloseable` and therefore can be used in a try-resource clause.
- [.NET, C, Java] **Removed** deprecated method `PaintForm`.
- [.NET, C, Java] **Changed** error behavior of method `PaintGroup`: A `Conformance` error is generated when trying to paint a group transparently and the requested document conformance does not support transparency.

Interface Document

- [.NET, C, Java] **New** property `Permissions` for extracting the permission settings of encrypted documents.
- [.NET, C, Java] **New** property `ViewerSettings`.
- [.NET, C, Java] **New** method `CopyAnnotation`.
- [.NET, C, Java] **New** method `CopyFileReference`.
- [.NET, C, Java] **New** method `CopyGroupElementWithoutContent` to facilitate recursive content copying.
- [.NET, C, Java] **New** method `CopyOutlineItem`.
- [.NET, C, Java] **New** method `CopyViewerSettings`.
- [.NET, C, Java] **New** method `CreateFileAttachmentAnnotation`.
- [.NET, C, Java] **New** method `CreateFreeTextAnnotation`.
- [.NET, C, Java] **New** method `CreateStickyNoteAnnotation`.
- [.NET, C, Java] **New** method `CreateTextStampAnnotation`.
- [.NET, C, Java] **New** method `CreateCircleAnnotation`.
- [.NET, C, Java] **New** method `CreateSquareAnnotation`.
- [.NET, C, Java] **New** method `CreateLineAnnotation`.
- [.NET, C, Java] **New** method `CreatePolyLineAnnotation`.
- [.NET, C, Java] **New** method `CreatePolygonAnnotation`.
- [.NET, C, Java] **New** method `CreateFreeDrawingAnnotation`.
- [.NET, C, Java] **Changed** error behavior of method `CreateFileReference`: A `Conformance` error is generated if the document's conformance does not allow the creation.
- [Java] **Changed** inheritance. Interface now inherits from `AutoCloseable` and therefore can be used in a try-resource clause.
- [.NET, C, Java] **Removed** deprecated methods `CreateForm` and `CopyPageAsForm`.
- [.NET, C, Java] **Changed** handling of conformance in methods `CopyColorSpace`, `CopyMetadata`, `CopyContentElement`, `CopyFormFieldNode`: The conformance of the target document is adapted automatically and if that's not possible, an error is generated.

Interface Image

- [.NET, C, Java] **Removed** deprecated property `IsMask`.

Interface PathGenerator

- [Java] **Changed** inheritance. Interface now inherits from `AutoCloseable` and therefore can be used in a try-resource clause.

Interface TextGenerator

- [Java] **Changed** inheritance. Interface now inherits from `AutoCloseable` and therefore can be used in a try-resource clause.

Interface `TextField`

- [.NET, C, Java] **New** property `FontSize`.

7.3 Changes in Version 5

- **New** additional supported operating system: Windows Server 2019.
- [.NET] **Changed**`PdfToolboxNET.dll` library. Cross-product functionality is outsourced into common library `PdfCommonNET.dll`.
- [.NET, C, Java] **New** interfaces `FileReference` and `FileReferenceList` for embedded and associated files.
- [.NET, C, Java] **New** enumeration `TextAlignment`.
- [.NET, C, Java] **New** interfaces for extracting, creating, and filling out AcroForm form fields:
  - `FormFieldNodeMap`
  - `AnnotationList`
  - `WidgetList`
  - `Annotation`
  - `Widget`
  - `FormFieldNode`
  - `SubForm`
  - `FormField`
  - `TextField`
  - `GeneralTextField`
  - `CombTextField`
  - `PushButtonField`
  - `CheckBoxField`
  - `RadioButton`
  - `RadioButtonList`
  - `RadioButtonField`
  - `ChoiceItem`
  - `ChoiceItemList`
  - `ChoiceField`
  - `ListBoxField`
  - `ComboBoxField`
- **New** interface `StringMap` for supporting custom metadata entries.
- [.NET, C, Java] **New** interfaces for extracting and creating outline items:
  - `OutlineItem`
  - `OutlineItemList`
  - `Destination`
  - `NamedDestination`
  - `DirectDestination`
  - `LocationZoomDestination`
  - `FitPageDestination`
  - `FitWidthDestination`
  - `FitHeightDestination`
  - `FitRectangleDestination`
- **New** license feature `Annotation`.

Interface Document

- [.NET, C, Java] **New** property `AssociatedFiles` for adding associated files.
- [.NET, C, Java] **New** property `EmbeddedFiles` for extracting and adding embedded files.
- [.NET, C, Java] **New** property `FormFields` for extracting form fields.
- [.NET, C, Java] **New** property `OpenDestination` for extracting and setting the destination that is displayed when opening the file.
- [.NET, C, Java] **New** property `OutlineItems` for extracting and adding outline items.
- [.NET, C, Java] **New** method `CreateFileReference` for creating embedded or associated files.
- [.NET, C, Java] **New** method `CreateSubForm` for creating sub forms.
- [.NET, C, Java] **New** method `CreateGeneralField` for creating general text form fields.
- [.NET, C, Java] **New** method `CreateCombField` for creating comb text form fields.
- [.NET, C, Java] **New** method `CreateCheckBoxField` for creating check box form fields.
- [.NET, C, Java] **New** method `CreateRadioButtonField` for creating radio button form fields.
- [.NET, C, Java] **New** method `CreateListBoxField` for creating list box form fields.
- [.NET, C, Java] **New** method `CreateComboBoxField` for creating combo box form fields.
- [.NET, C, Java] **New** method `CreateField` for creating form fields.
- [.NET, C, Java] **New** method `CreateNamedDestination` for creating new named destinations.
- [.NET, C, Java] **New** method `CreateOutlineItem` for creating new outline items.
- [.NET, C, Java] **New** method `CopyFormFieldNode` for filling out form fields.

Interface Metadata

- [.NET] **Deprecated** property `CreationDate`.
- [.NET] **New** property `CreationDate2` supersedes `CreationDate`.
- [.NET] **Deprecated** property `ModificationDate`.
- [.NET] **New** property `ModificationDate2` supersedes `ModificationDate`.
- [.NET, C, Java] **New** property `CustomEntries` to access custom document info entries.

Interface Page

- [.NET, C, Java] **New** property `Annotations` for extracting annotations and form field widgets.

Interface TextFragment

- [.NET, C, Java] **Removed** property `Transformation`.
- [.NET, C, Java] **New** property `Transform` to replace removed property `Transformation`.

7.4 Changes in Version 4.12

- **New** support for encryption according to PDF 2.0 (revision 6, replaces deprecated revision 5).
- **New** HTTP proxy setting in the GUI license manager.
- **Introduced** license features `Assembly`, `Creation`, `Extraction`, `Annotation`, and `Modification`.
- [.NET] **Corrected** enum `CopyOption`: Removed wrongly introduced element `CopyOptionalContentgroups` and changed values of all subsequent elements.
- [.NET] **New** enum elements `CopyAssociatedFiles`, `MergeOCGs`, and `SeparateAcroForms` in enum `CopyOption`.

- [Java] **New** enum elements `COPY_ASSOCIATED_FILES`, `MERGE_OCGS`, and `SEPERATE_ACROFORMS` in enum `CopyOption`.
- [.NET, Java] **Changed** behavior of property getters to signal errors instead of just returning default values.
- [.NET, C, Java] **New** interface `ImageMask` for images that are used as stencil masks.
- [.NET, C, Java] **Deprecated** interface `Form` (superseded by `Group`).
- [.NET, C, Java] **New** interface `Group` for transparency group resources.
- [.NET, C, Java] **New** struct `TransparencyParams` that groups transparency related parameters.
- [.NET, C, Java] **New** enum `ColorSpaceType` that denotes the type of a color space.
- [.NET, C, Java] **New** struct `FillParams` that groups together fill parameters for paths and text.
- **New** feature: Page content extraction and modification through the following new interface elements:
  - [.NET, C, Java] **New** interface `ContentExtractor`.
  - [.NET, C, Java] **New** enum `UngroupingSet`.
  - [.NET, C, Java] **New** interface `TextFragment`.
  - [.NET, C, Java] **New** interface `ContentElement`.
  - [.NET, C, Java] **New** interface `TextElement`.
  - [.NET, C, Java] **New** interface `GroupElement`.
  - [.NET, C, Java] **New** interface `PathElement`.
  - [.NET, C, Java] **New** interface `ImageElement`.
  - [.NET, C, Java] **New** interface `ImageMaskElement`.
  - [.NET, C, Java] **New** interface `ShadingElement`.

Interface Document

- [.NET, C, Java] **Deprecated** method `CreateForm` (superseded by `CreateGroup`).
- [.NET, C, Java] **New** method `CreateGroup` to create a new transparency group resource.
- [.NET, C, Java] **New** method `CreateImageMask` to create a new stencil mask resource.
- [.NET, C, Java] **Deprecated** method `CopyPageAsForm` (superseded by `CopyPageAsGroup`).
- [.NET, C, Java] **New** method `CopyPageAsGroup` to create a transparency group resource from an input page.
- [.NET, C, Java] **New** method `CopyContentElement` to copy an input content element to an output document.

Interface ColorSpace

- [.NET, C, Java] **New** property `Type`.

Interface Transformation

- [.NET, C, Java] **New** copy constructor.
- [.NET, C, Java] **New** method `Invert`.
- [.NET, C, Java] **New** method `TransformPoint`.

Interface ContentGenerator

- [.NET, C, Java] **Deprecated** method `PaintForm` (superseded by `PaintGroup`).
- [.NET, C, Java] **New** method `PaintGroup` to place a transparency group resource onto the output content.
- [.NET, C, Java] **Changed** method `PaintImageMask`: The type of the image to be painted as image mask is changed from `Image` to `ImageMask`.
- [.NET, C, Java] **New** method `AppendContentElement` to place a copied input content element onto the output content.

Interface Paint

- [.NET, C, Java] **New** property ColorSpace.
- [.NET, C, Java] **New** property Color.
- [.NET, C, Java] **New** property Transparency.

Interface Image

- [.NET, C, Java] **Deprecated** property IsMask (superseded by separate ImageMask interface).

Interface Text

- [.NET, C, Java] **New** iterator interface for extracting TextFragments.

7.5 Changes in Version 4.11

- **New** support for reading and writing PDF 2.0 documents.
- **New** support for the creation of output files larger than 10GB (not PDF/A-1).
- **Improved** search in installed font collection to also find fonts by other names than TrueType or PostScript names.
- **Improved** font subsetting of CFF and OpenType fonts.
- **New** optimization of output file size for documents that contain structure information.

Interface Document

- [.NET, C, Java] **Changed** method CreateDeviceColorSpace: PDF/A conformance of the document is now removed if not explicitly requested and if no output intent is set.

Enumeration Conformance

- [.NET, C, Java] **New** item Pdf20.

7.6 Changes in Version 4.10

- **Improved** robustness against corrupt input PDF documents.
- [C] **Clarified** Error handling of TPdfStreamDescriptor functions.
- **Improved** reparation of corrupt form fields.
- **New** support for writing PDF objects into object streams. Most objects that are contained in object streams in the input document are now also stored in object streams in the output document. When enabling linearization, however, no objects are stored in object streams.
- [.NET, C, Java] **New** enumeration BlendMode.

Interface Document

- [.NET, C, Java] **Changed** the error handling of method Open:
  - Opening a PDF collection (portfolio) will now result in an error of type UnsupportedFeature.

  - Opening a PDF with an unsupported PDF version will now result in an error of type `Conformance`.
- [.NET, C, Java] **New** method `CreateMetadata` to create new metadata objects.
- [.NET, C, Java] **Changed** the behavior of method `CopyPage` and `CopyPageAsForm`:
  - The outline structure in the output file now always matches the outline structure in the input file, regardless of the order in which pages are copied.
  - If the conformance of the output document is explicitly set to PDF/A level A and the copy option `CopyLogicalStructure` is not set, an error of type `Conformance` is signaled.
- [.NET, C, Java] **New** method `CreateBlendingPaint` to create a paint with a specific blend mode.

Interface Page

- [.NET, C, Java] **New** property `Metadata` to access or modify the page metadata.
- [.NET, C, Java] **New** property `MediaBox` to access the page's media box.
- [.NET, C, Java] **New** property `BleedBox` to access the page's bleed box.
- [.NET, C, Java] **New** property `TrimBox` to access the page's trim box.
- [.NET, C, Java] **New** property `ArtBox` to access the page's art box.

Interface `Metadata`

- [.NET, C, Java] **New** property `Xmp` to access the XMP metadata stream.

7.7 Changes in Version 4.9

- **Improved** support for and robustness against corrupt input PDF documents.
- **Improved** repair of embedded font programs that are corrupt.
- **New** support for OpenType font collections in installed font collection.
- **Improved** metadata generation for standard PDF properties.
- [C] **Changed** return value `pfGetLength` of `TPDFStreamDescriptor` to `pos_t`[6].
- [Java] **New**: MemoryStream implementation for Stream interface.

Interface `Document`

- [.NET, C, Java] **New** method `CreateAlphaPaint`: Create transparent paint using alpha blending that can be used for filling and stroking paths and text.
- [.NET, C, Java] **Changed** method `Create`: The behavior of the parameter `conformance` is now more consistent and predictable.

  Previous behavior:
  - If an explicit conformance was given, the value was changed silently if content from an incompatible document was added later.
  - If no explicit conformance was given, using certain features would still lead to an error, depending on the currently determined conformance level.

  New behavior:
  - If an explicit conformance is given, the value is always respected.
  - If no explicit conformance is given, the value will always be determined automatically and no errors will occur.

[6] This has no effect on neither the .NET, Java, nor COM API

- [.NET, C, Java] **Changed** behavior of methods `CopyPage` and `CopyPageAsForm`:
  These methods will now report an error when trying to copy from a document with a conformance incompatible to the conformance specified in the method `Create`.
- [.NET, C, Java] **Changed** behavior of methods `CreateFont` and `CreateSystemFont`:
  These methods will no longer report an error for non-embedded fonts, if the conformance was **automatically** determined to be PDF/A. The conformance is now changed to a suitable value instead.
  If the conformance was explicitly specified, the error is still reported.

Interface `Transformation`

- [.NET, C, Java] **New** method `RotateAround` to rotate around a specific point.

7.8 Changes in Version 4.8

- **Improved** creation of annotation appearances to use less memory and processing time.
- **Added** repair functionality for TrueType font programs whose glyphs are not ordered correctly.
- [.NET, C, Java] **New** interface `SignatureField` gives access to certain properties of a signature field.
- [C] **Renamed** struct `TPdfRect` to `TPdfRectangle` for consistency reasons.
- [.NET] **Renamed** struct `Rect` to `Rectangle` for consistency reasons.

All interfaces

- [.NET, C, Java] **New** static property `ProductVersion` to identify the product version.
- [.NET] **New** static property setter `LicenseKey`.
- [.NET, Java] **Changed** all property setters to allow throwing exceptions.
- [C] **Changed** all property setters to return a `BOOL` return value.

Interface `Document`

- [.NET, C, Java] **New** property `SignatureFields` to enumerate the signature fields of the document.
- [.NET, C, Java] **New** property setter for property `Metadata`.
- [.NET, C, Java] **New** function `CopyMetadata`.

Interface Page

- [.NET] **Changed** method `Rotate` to take a parameter of type `Rotation` instead of `int` for consistency reasons.

Interface `PageList`

- [C] **Renamed** all function prefixes from `PdfPages…` to `PdfPageList…` for consistency reasons.

Interface Page

- [.NET, C, Java] **Removed** property `Mask`, as it was barely useful in its current form.

Struct `Size`

- [Java] **Renamed** field `w` to `width` for consistency reasons.
- [Java] **Renamed** field `h` to `height` for consistency reasons.

Enum `Conformance`

- [C] **Renamed** all value prefixes from `ePDF…` to `ePdf…` for consistency reasons.

8 Licensing, Copyright, and Contact

PDF Tools AG is a world leader in PDF (Portable Document Format) software, delivering reliable PDF products to international customers in all market segments.

PDF Tools AG provides server-based software products designed specifically for developers, integrators, consultants, customizing specialists and IT-departments. Thousands of companies worldwide use our products directly and hundreds of thousands of users benefit from the technology indirectly via a global network of OEM partners. The tools can be easily embedded into application programs and are available for a multitude of operating system platforms.

Licensing and Copyright The 3-Heights® PDF Toolbox API is copyrighted. This user's manual is also copyright protected; It may be copied and given away provided that it remains unchanged including the copyright notice.

Contact

PDF Tools AG
Brown-Boveri-Strasse 5
8050 Zürich
Switzerland
http://www.pdf-tools.com
pdfsales@pdf-tools.com