PDF-XChange OCR SDK

User Manual

Head Quarters:
Tracker Software Products (Canada) Ltd.,

P.O. Box 79, 9622 Chemainus Road
Chemainus. V0R 1K0
British Columbia, Canada

Sales
Tel: Canada (+00) 1-250-324-1621
Fax: Canada (+00) 1-250-324-1623

In Europe:

Unit 17, Raleigh Court. Priestly Way.
Crawley. Sussex. RH10 9PD.
England.

Sales
Tel: +44 (0) 20 8555 1122
Fax: (+00) 1-250-324-1623

http://www.tracker-software.com

sales@tracker-software.com

Created: Thursday, April 17, 2014

Copyright © 2001-2014 Tracker Software Products. All Rights Reserved.

PDF-X OCR SDK

copyright © Tracker Software Products. All rights reserved.
http://www.tracker-software.com

The information contained in this document is subject to change without notice.
This document contains proprietary information which is protected by copyright.
All rights are reserved. No part of this document may be photocopied, reproduced,
or translated to another language without the prior written consent of Tracker Software Products.

Table of Contents

PDF-X OCR SDK

Online **Help**

Welcome to the PDF-X OCR SDK online help. Browse through the help pages by clicking on the icons below or selecting pages in the table of contents to the left. To quickly find specific product information, enter search criteria in the search box above and click the search button.

Getting Started
OCR Library Types
Input List Handling

High-Level Functions
Low-Level Functions
Error Handling

Tutorials
FAQ

Ask Us

If you're unable to find what you're looking for in this help system, try these alternative resources:

- Our Website
- Knowledgebase
- FAQ

or contact our support team:
Email: support@tracker-software.com

Most popular pages

- PDF-X OCR SDK
- OCR_Init
- OCR_NewPagelist
- Getting Started
- PXO_InputField
- FAQ
- PXO_InputFields
- OCR_PagesToInputFields
- OCR Library Types
- OCR_MakeSearchable

Home > Getting Started

Getting Started

Articles in this section

Introduction   System Requirements

See also

PDF-X OCR SDK
OCR Library Types
Input List Handling
High-Level Functions
Low-Level Functions
Error Handling
Tutorials
FAQ

Home > Getting Started > Introduction

Introduction

Introduction to PDF-X OCR SDK

PDF-X OCR SDK– Version 1.x

The PDF-X OCR SDK is available with our PDF-Tools PRO SDK. See the included license file for full details of the terms of use.

This toolkit as with all our developer kits may not be used to develop Toolkits or Components of any type for use by other non-licensed developers or for use to assist in the creation of Printer driver under the usual license conditions provided. For more information on licensing please read the license agreement and if any doubt as to whether your intended use would be in breach of the license terms please contact us to discuss your needs in more depth as we do offer alternate licensing and will tailor our agreement to meet your needs in most circumstances under different terms of supply.

Support

Support is available direct from our user forums **http://www.tracker-software.com/forum/index.php**.

We recommend that developers use the evaluation download as extensively as possible prior to purchase. The evaluation version is fully functional but will only OCR the first two pages of any PDF document – only on purchase will you be provided with the serial number and unlock string required by each component to be passed within your application code (see the demo applications provided) to enable PDF generation without this demo watermark stamp.

By virtually creating your application to the point where you are ready for distribution before you purchase, you are guaranteed satisfaction and we do not disappoint developers asking for refunds once purchased – we do not offer any money back options – so please ensure you are 100% satisfied before you purchase.

Developers may also find it useful when developing applications for the purpose of creating and manipulating Adobe PDF Formats - to download the documentation relevant to this format from the Adobe web site - at the time of writing this is currently free and may prove useful in explaining in more detail the functionality available. **http://www.adobe.com/**.

Tracker Software Products Ltd also provides End User and Developer Tool Kits for the creation and manipulation of PDF and Raster Image files and Virtual Printer Drivers. For more information please visit **http://www.tracker-software.com**.

License Agreement

License Agreement PDF-XChange Viewer (SDK) from Tracker Software Products Ltd 2006-2012.

Please see the license agreement included with the distribution for license information (PDF-XChange_OCR_SDK_License.pdf)

Redistribution of the PDF-X OCR SDK

Redistribution of PDF-X OCR SDK components

The PDF-X OCR SDK depends only on the ocrtools.dll, and is not reliant on any other PDF-XChange/Tools Image-XChange SDK components. However, please note, The PDF-XChange components take advantage of the Microsoft© GDI+ for vector printing and it is required to have installed it on the OS where it is not installed by default (all Windows prior to Windows XP). The PDF-X OCR SDK is available for Windows 2000 and later only - earlier versions of Windows are not supported.

See also

System Requirements

Home > Getting Started > System Requirements

System Requirements

PDF-XChange Pro 2012 supports all Windows (32/64 bit) operating systems from Windows XP and later.

Version 5 (2012): Microsoft/Citrix Terminal Server compatible*****.
Version 4: Microsoft/Citrix Terminal Server compatible*****.
Version 3: Still available for Windows 95 and later.

*** Note: Though many users have virtualized some of our component products such as the PDF-XChange Viewer and PDF-Tools application using XenApp, we do not support this at this time.**
Particularly the printer drivers are not designed to work in a virtualized environment.

See also

Introduction

Home > OCR Library Types

OCR Library Types

This opening paragraph should describe the feature that you are documenting. Explain how it is commonly used and what the benefits are. For example: The Widget Master email link allows you to easily send information about each widget to various departments within your company. Often, the feature that you are documenting can be best explained by walking the reader through step by step. Use screenshots to illustrate the steps where possible.

1. Start the application by...
2. On the startup screen, click the...
3.

See also

PDF-X OCR SDK
Getting Started
Input List Handling
High-Level Functions
Low-Level Functions
Error Handling
Tutorials
FAQ

Home > OCR Library Types > PXO_Pagelist

PXO_Pagelist

Input type used to store PDF page numbers (zero-indexed) for OCR operations such as **OCR_MakeSearchable**. This type is a typedef to a void pointer, which is used to store the address of an internal std::vector-like structure. Memory used by this is released with **OCR_ReleasePagelist**.

See also

PXO_InputField
PXO_InputFields
PXODocument
PXO_Options
PXO_Language
PXO_CallbackStage
OCR_RegionMode
OCR_ImageProcessingFlags
OCR_RasterPageSettings
OCR_Baseline
OCR_SymbolBox
PXO_Page
PXO_Region
PXO_FieldInputFlags

Home > OCR Library Types > PXO_InputField

PXO_InputField

Input structure for regional / zonal OCR. Coordinates are in PDF page coordinates, starting with (0,0) at the lower left corner of the page, similar to conventional / Cartesian mathematics (and in contrast to Windows RECT coordinates). The unit is points (72 per inch).

MEMBER	TYPE	MEANING
`left`	`double`	Left boundary of input field rectangle on page.
`bottom`	`double`	Bottom boundary of input field rectangle on page.
`right`	`double`	Right boundary of input field rectangle on page.
`top`	`double`	Top boundary of input field rectangle on page.
`nPage`	`DWORD`	The page number of the input PDF on which to OCR the zonal region.

`whitelist`	BSTR	Optional character whitelist. Can be NULL (no whitelist).
`blacklist`	BSTR	Optional character blacklist. Can be NULL (no blacklist).
`label`	BSTR	Optional field label. Not used internally yet, but may be used to track the meaning of OCR output from a particular field (e.g. a database field name) for other development in your application.

NOTE: BSTR variables must be allocated with SysAllocString() and deallocated with SysFreeString(). All SDK functions that take PXO_InputField structures as input store BSTR members as copies internally, when appropriate. Likewise, SDK functions that return a PXO_InputField structure allocate new BSTR for the relevant members which must be deallocated with SysFreeString().

See also

PXO_Pagelist
PXO_InputFields
PXODocument
PXO_Options
PXO_Language
PXO_CallbackStage
OCR_RegionMode
OCR_ImageProcessingFlags
OCR_RasterPageSettings
OCR_Baseline
OCR_SymbolBox
PXO_Page
PXO_Region
PXO_FieldInputFlags

Home > OCR Library Types > PXO_InputFields

PXO_InputFields

Input structure containing a list of **PXO_InputField** structs for zonal / regional OCR.

See also

PXO_Pagelist
PXO_InputField
PXODocument
PXO_Options
PXO_Language
PXO_CallbackStage
OCR_RegionMode
OCR_ImageProcessingFlags
OCR_RasterPageSettings
OCR_Baseline
OCR_SymbolBox
PXO_Page
PXO_Region
PXO_FieldInputFlags

Home > OCR Library Types > PXODocument

PXODocument

The main document structure used by the PDF-X OCR SDK. Upon loading with **OCR_LoadW**, contains an input layer. After a call to **OCR_MakeSearchable**, it will also contain a separate output document layer, which can be saved with **OCR_SaveW**.

See also

PXO_Pagelist
PXO_InputField
PXO_InputFields
PXO_Options
PXO_Language
PXO_CallbackStage
OCR_RegionMode
OCR_ImageProcessingFlags
OCR_RasterPageSettings
OCR_Baseline

OCR_SymbolBox
PXO_Page
PXO_Region
PXO_FieldInputFlags

PXO_Options

OCR options input structure.

MEMBER	TYPE	MEANING
`lang`	`PXO_Language`	OCR language identifier; ensure language is installed in correct location. See http://www.tracker-software.com for additional OCR language pack downloads, or email support@tracker-software.com
`RegionMode`	`OCR_RegionMode`	A region mode specifier. Useful for increasing OCR accuracy and speed where the type of input is known ahead of time, for example a single line or a single paragraph / block of text. For most common full-page OCR tasks, OCR_Auto is an appropriate value.
`whitelist`	`BSTR`	A list of character to accept as recognizable symbols. All others will be ignored. Useful for increasing accuracy when input type is known.
`blacklist`	`BSTR`	A list of characters to deny as acceptable symbols for recognition. All others will be considered suitable candidates for OCR identification. Useful for removing unlikely symbols when the input type is known.
`DataPath`	`BSTR`	The path to the language pack directory containing the subfolder **ocrdats/**. For example, if languages are installed in **C:\OCR_Application\Languages\ocrdats\**, this member will be assigned the string **"C:\OCR_Application\Languages\"**. Pointing it at the **ocrdats** folder itself will result in an error; it must be the parent directory containing this folder. Attempting OCR with an incorrect language directory will result in an error being returned.
`ImageFlags`	`DWORD`	Flags for image processing. See **OCR_ImageProcessingFlags**.
`raster_dpi`	`int`	DPI setting for rasterizing / resampling pages for OCR. The OCR algorithm takes the entire page as input to avoid complications resulting from fragmented images and multiple images on a single page. A good value for high accuracy is 300 DPI. 150 DPI may introduce slightly more errors but with possibly improved recognition speed. The effect of this setting is ultimately limited by the resolution of embedded images in the PDF, and there is currently no check to ensure images are not being needlessly up-sampled.
`accMode`	`int`	Reserved for future use. Please set to zero.

See also

PXO_Pagelist
PXO_InputField
PXO_InputFields
PXODocument
PXO_Language
PXO_CallbackStage
OCR_RegionMode
OCR_ImageProcessingFlags
OCR_RasterPageSettings
OCR_Baseline
OCR_SymbolBox
PXO_Page
PXO_Region
PXO_FieldInputFlags

PXO_Language

An enum containing the full list of language possibilities (see **ocr_types.h** for a full list). Please ensure you have installed the correct language pack(s) or data file(s) for the language you select. See **PXO_Options**.

See also

PXO_Pagelist
PXO_InputField
PXO_InputFields
PXODocument
PXO_Options
PXO_CallbackStage
OCR_RegionMode

OCR_ImageProcessingFlags
OCR_RasterPageSettings
OCR_Baseline
OCR_SymbolBox
PXO_Page
PXO_Region
PXO_FieldInputFlags

Home > OCR Library Types > PXO_CallbackStage

PXO_CallbackStage

An enum passed to the user-defined callback function set by **OCR_SetCallBack**.

CONSTANT	VALUE	MEANING OF dwLevel
`PXOClb_Start`	1	`MaxVal` - maximum value of the level which will be passed
`PXOClb_Processing`	2	Current progress level - any value from `0` to `MaxVal`
`PXOClb_Finish`	3	May be any value from `0` to `MaxVal` (`MaxVal` if all passed), may be ignored

See also

PXO_Pagelist
PXO_InputField
PXO_InputFields
PXODocument
PXO_Options
PXO_Language
OCR_RegionMode
OCR_ImageProcessingFlags
OCR_RasterPageSettings
OCR_Baseline
OCR_SymbolBox
PXO_Page
PXO_Region
PXO_FieldInputFlags

Home > OCR Library Types > OCR_RegionMode

OCR_RegionMode

One of the first stages of OCR is page segmentation, through which the layout of text lines and paragraphs is determined. Setting the region mode is one way to improve the accuracy of page segmentation when the layout is known ahead of time (for example with zonal OCR where single lines or words are being selected for recognition). In most cases of full page OCR, automatic layout analysis is best.

CONSTANT	VALUE	MEANING
`OCR_Auto`	`1`	Automatically determine text layout – best for most tasks for unknown input data.
`OCR_SingleColumn`	`4`	A single column of text.
`OCR_VerticalText`	`5`	Vertical text (horizontal upright characters arranged in a vertical line).
`OCR_Block`	`6`	A block of text (paragraph).
`OCR_Line`	`7`	A single line of input text.
`OCR_Word`	`8`	A single word.
`OCR_Symbol`	`10`	One character.
`OCR_AutoRotateImageOnly`	`999`	Special flag used only with **OCR_MakeSearchable** – specifies that recognition is to be used only to straighten / automatically rotate the input pages and output them to a new PDF without performing OCR. The new document will be a copy of the original (or the subset specified by the input page list) with the pages reoriented to best fit horizontal lines of text to the horizontal. This can be useful for pre-processing an image-based PDF prior to performing some other task (e.g. zonal OCR).

See also

PXO_Pagelist
PXO_InputField
PXO_InputFields
PXODocument

PXO_Options
PXO_Language
PXO_CallbackStage
OCR_ImageProcessingFlags
OCR_RasterPageSettings
OCR_Baseline
OCR_SymbolBox
PXO_Page
PXO_Region
PXO_FieldInputFlags

Home > OCR Library Types > OCR_ImageProcessingFlags

OCR_ImageProcessingFlags

CONSTANT	VALUE	MEANING
`OCR_Image_NoRotate`	`0x0000`	Do not automatically rotate input images; OCR as-is.
`OCR_Image_Autorotate`	`0x0001`	Automatically rotate input images. This functionality is meant to correct minor skews (up to 45 degrees) in the page orientation resulting from slightly non-square scanning. Please ensure pages are input right-side-up before running OCR. Future releases may add full orientation detection.
`OCR_Image_EdgeRefine`	`0x0002`	Smart blur method for reducing letter edge artifacts (good for upsampled images). Not recommended for less than 300 DPI resolution setting.
`OCR_Image_GaussianBlur`	`0x0004`	Simple blur method for reducing letter edge artifacts.

See also

PXO_Pagelist
PXO_InputField
PXO_InputFields
PXODocument
PXO_Options
PXO_Language
PXO_CallbackStage
OCR_RegionMode
OCR_RasterPageSettings
OCR_Baseline
OCR_SymbolBox
PXO_Page
PXO_Region
PXO_FieldInputFlags

Home > OCR Library Types > OCR_RasterPageSettings

OCR_RasterPageSettings

This structure is used by some low level SDK functions to assist with transforming PDF coordinates to rasterized page image coordinates and vice versa. It is returned by **OCRp_Page** and **OCRp_Field** and used by **OCRp_RasterRectToPDF**.

MEMBER	TYPE	MEANING
`imgDPI`	`int`	DPI of the rasterized page image used by the internal OCR engine.
`scalefactor`	`double`	A factor for converting (scaling) the page size and coordinates in points to the rasterized page size and coordinates in DPI. It is equal to the input DPI / 72 pts. Conversion from PDF coordinates to rasterized page image coordinates is done by multiplying by the scale factor; the inverse is done by dividing by the scale factor.
`pdfwidth`	`double`	Width of the PDF page in points.
`pdfheight`	`double`	Height of the PDF page in points.
`imgwidth`	`int`	Width of the rasterized image in pixels.
`imgheight`	`int`	Height of the rasterized image in pixels.

See also

PXO_Pagelist
PXO_InputField
PXO_InputFields
PXODocument
PXO_Options

PXO_Language
PXO_CallbackStage
OCR_RegionMode
OCR_ImageProcessingFlags
OCR_Baseline
OCR_SymbolBox
PXO_Page
PXO_Region
PXO_FieldInputFlags

Home > OCR Library Types > OCR_Baseline

OCR_Baseline

Structure for storing the baseline for a text element such as a line of text. Also includes a height field (for the height of the text element). These are coordinates in rasterized-image page coordinates. (X1,Y1) is the left hand point of the line segment. (X2, Y2) is the right-hand point of the line segment. Height is the height of the text element.

MEMBER	TYPE	MEANING
`X1`	`int`	Coordinates in raster image coordinates (pixels – (0,0) at top right of page, as per Windows RECT coordinates).
`X2`	`int`	
`Y1`	`int`	
`Y2`	`int`	
`Height`	`int`	Average height of the text element; may not always be used.

See also

PXO_Pagelist
PXO_InputField
PXO_InputFields
PXODocument
PXO_Options
PXO_Language
PXO_CallbackStage
OCR_RegionMode
OCR_ImageProcessingFlags
OCR_RasterPageSettings
OCR_SymbolBox
PXO_Page
PXO_Region
PXO_FieldInputFlags

Home > OCR Library Types > OCR_SymbolBox

OCR_SymbolBox

A structure containing a single character and descriptive information from OCR (when available). Underlined fields are still experimental and should be used with caution.

MEMBER	TYPE	MEANING
`rcBound`	`RECT`	Rectangular bounding region of the symbol in rasterized page image coordinates.
`LineBaseline`	`OCR_Baseline`	Baseline for the **line** containing the symbol. Useful for aligning symbols precisely along a text line.
`rcLineBound`	`RECT`	Rectangular bounding region for the symbol’s enclosing **line**. For single symbols recognized outside of the context of a surrounding line it is equivalent to the symbol’s bounding rectangle.
`wcSymbol`	`wchar_t`	Unicode symbol recognized by OCR.
`nConfidence`	`float`	A confidence estimation. Higher values are better confidence.*
`isbold`	`bool`	Bold? True / false
`isitalic`	`bool`	Italic? True / false
`isunderlined`	`bool`	Underlined? True / false
`isserif`	`bool`	Serif font? True / false
`ismonospace`	`bool`	Monospace font? True / false
`issmallcaps`	`bool`	Small caps font? True / false

`pointsize`	`SHORT`	Font size estimate in points. Reasonably accurate; bounding boxes may also be used for size estimation.
`fontid`	`SHORT`	Reserved; unused.

See also

- PXO_Pagelist
- PXO_InputField
- PXO_InputFields
- PXODocument
- PXO_Options
- PXO_Language
- PXO_CallbackStage
- OCR_RegionMode
- OCR_ImageProcessingFlags
- OCR_RasterPageSettings
- OCR_Baseline
- PXO_Page
- PXO_Region
- PXO_FieldInputFlags

Home > OCR Library Types > PXO_Page

PXO_Page

OCR results are arranged in a hierarchy. **PXO_Page** is the top level of the hierarchy, and may contain one or more **PXO_Region** members, accessed by **OCRp_GetRegionFromPage**. Memory must be freed with **OCRp_FreePage** when finished (this will destroy all associated PXO_Regions as well).

See also

- PXO_Pagelist
- PXO_InputField
- PXO_InputFields
- PXODocument
- PXO_Options
- PXO_Language
- PXO_CallbackStage
- OCR_RegionMode
- OCR_ImageProcessingFlags
- OCR_RasterPageSettings
- OCR_Baseline
- OCR_SymbolBox
- PXO_Region
- PXO_FieldInputFlags

Home > OCR Library Types > PXO_Region

PXO_Region

The second level in the OCR results hierarchy, this may contain one or more **OCR_SymbolBox** members, accessible with the **OCRp_GetSymbolFromRegion** function. Regions are determined by initial page analysis and the exact meaning of a region depends significantly on the layout of the page. Thus there is no general definition of a region that applies universally, except that it is a subset of the whole page containing some spatially continuous text. In some cases there may be only one region in a page (e.g. if it contains only one large block of text).

Note the difference between an output region (**PXO_Region**) and an input field (**PXO_InputField**). An input field is a rectangular area of the page specified in PDF page coordinates within which OCR results are calculated. An output region (**PXO_Region**) is simply a subset of the output results that forms a continuous area of text. After OCR, the **PXO_Page** results structure returned from **OCRp_Field** may contain one or more output **PXO_Region**(s), depending on the text layout found inside the input field.

See also

- PXO_Pagelist
- PXO_InputField
- PXO_InputFields
- PXODocument
- PXO_Options
- PXO_Language
- PXO_CallbackStage
- OCR_RegionMode

OCR_ImageProcessingFlags
OCR_RasterPageSettings
OCR_Baseline
OCR_SymbolBox
PXO_Page
PXO_FieldInputFlags

Home > OCR Library Types > PXO_FieldInputFlags

PXO_FieldInputFlags

This **enum** specificies flags used to denote the style of coordinates used in OCR input fields **PXO_InputField**. It impacts how the numerical values of the "bottom" and "top" components of the coordinates are interpreted relative to the page. The default, `PXO_Origin_BottomLeft`, is how the PDF-X OCR SDK interpreted fields prior to version 1.0.5 of the DLL in which this change was introduced.

CONSTANT	VALUE	MEANING
`PXO_Origin_BottomLeft`	`0x0000`	Default; use the bottom left corner of the PDF page as the origin, with y increasing towards the top of the page. This is the standard mathematical Cartesian coordinate system, as used internally by PDF documents.
`PXO_Origin_TopLeft`	`0x0001`	Fields use the top left corner of the PDF page as the origin, with y increasing towards the bottom of the page. This is the style of coordinate axes used by the Windows API, and is also useful for assigning fields relative to the top of a page (which can help with ensuring fields are properly placed in a series of scanned documents which were placed on a flatbed scanner aligned by the top left corner).

See also

PXO_Pagelist
PXO_InputField
PXO_InputFields
PXODocument
PXO_Options
PXO_Language
PXO_CallbackStage
OCR_RegionMode
OCR_ImageProcessingFlags
OCR_RasterPageSettings
OCR_Baseline
OCR_SymbolBox
PXO_Page
PXO_Region

Home > Input List Handling

Input List Handling

This opening paragraph should describe the feature that you are documenting. Explain how it is commonly used and what the benefits are. For example: The Widget Master email link allows you to easily send information about each widget to various departments within your company. Often, the feature that you are documenting can be best explained by walking the reader through step by step. Use screenshots to illustrate the steps where possible.

1. Start the application by...
2. On the startup screen, click the...
3.

See also

PDF-X OCR SDK
Getting Started
OCR Library Types
High-Level Functions
Low-Level Functions
Error Handling
Tutorials
FAQ

Home > Input List Handling > OCR_NewPagelist

OCR_NewPagelist

OCR_NewPagelist internally initializes a new input page list (**PXO_Pagelist**) structure. You can think of this structure as something like an array or `std::vector class`, but all access is via API functions. The created structure should be deleted with **OCR_ReleasePagelist** when finished.

```
 HRESULT  OCR_NewPagelist(
      PXO_Pagelist *pPageList
  );
```

Parameters

pPageList pointer to a variable of the type PXO_Pagelist which will receive the created page-list object.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

See also

OCR_AddPage
OCR_NumPages
OCR_GetPageByIndex
OCR_ReleasePagelist
OCR_NewInputFields
OCR_AddInputField
OCR_NumInputFields
OCR_GetInputFieldByIndex
OCR_ReleaseInputFields
OCR_PagesToInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW
TEMPLATE FILES

Home > Input List Handling > OCR_AddPage

OCR_AddPage

OCR_AddPage adds a new input document page number to the **PXO_Pagelist** structure created by **OCR_NewPagelist**.

```
 HRESULT  OCR_AddPage(
    PXO_Pagelist PageList,
    DWORD nPage
);
```

Parameters

PageList PXO_Pagelist variable previously created by **OCR_NewPagelist**.

nPage A page number, zero-indexed, from the intended input document.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

See also

OCR_NewPagelist
OCR_NumPages
OCR_GetPageByIndex
OCR_ReleasePagelist
OCR_NewInputFields
OCR_AddInputField
OCR_NumInputFields
OCR_GetInputFieldByIndex
OCR_ReleaseInputFields
OCR_PagesToInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW
TEMPLATE FILES

Home > Input List Handling > OCR_NumPages

OCR_NumPages

OCR_NumPages returns the number of input page numbers currently stored in the **PXO_Pagelist** structure created by **OCR_NewPagelist**.

```
HRESULT  OCR_NumPages(
    PXO_Pagelist PageList,
    DWORD *nPages
);
```

Parameters

PageList PXO_Pagelist variable previously created by **OCR_NewPagelist**.

nPages Pointer to a DWORD, receives the number of input pages in the specified PXO_Pagelist structure.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

See also

OCR_NewPagelist
OCR_AddPage
OCR_GetPageByIndex
OCR_ReleasePagelist
OCR_NewInputFields
OCR_AddInputField
OCR_NumInputFields
OCR_GetInputFieldByIndex
OCR_ReleaseInputFields
OCR_PagesToInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW
TEMPLATE FILES

Home > Input List Handling > OCR_GetPageByIndex

OCR_GetPageByIndex

OCR_GetPageByIndex returns the specified input document page number from the **PXO_Pagelist** structure created by **OCR_NewPagelist**.

```
HRESULT  OCR_GetPageByIndex(
     PXO_Pagelist PageList,
     DWORD nIndex,
     DWORD *nPage
);
```

Parameters

PageList **PXO_Pagelist** variable previously created by **OCR_NewPagelist**.

nIndex Zero-indexed position (similar to an array index) of the input document page number to return from **PXO_Pagelist**. It must be less than the value returned by **OCR_NumPages**.

nPage Pointer to a DWORD, receives the specified page number stored in the **PXO_Pagelist** structure.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

Example (C++)

```
PXO_Pagelist inPages;
HRESULT hr;
DWORD nPages;
DWORD nPage;

hr = OCR_NewPagelist(&inPages);

OCR_AddPage(inPages, 1);
OCR_AddPage(inPages, 12);
OCR_AddPage(inPages, 17);

OCR_NumPages(inPages, &nPages);

for (DWORD i=0; i < nPages; i++)
{
  OCR_GetPageByIndex(PageList, i, &nPage);
    std::cout << "Index: " << i << ", Page number: " << nPage << std::endl;
}
// OUTPUT:
// Index: 0, Page number: 1
// Index: 1, Page number: 12
// Index: 2, Page number: 17

OCR_ReleasePagelist(&inPages);
```

See also

OCR_NewPagelist
OCR_AddPage
OCR_NumPages
OCR_ReleasePagelist
OCR_NewInputFields
OCR_AddInputField
OCR_NumInputFields
OCR_GetInputFieldByIndex
OCR_ReleaseInputFields
OCR_PagesToInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW
TEMPLATE FILES

Home > Input List Handling > OCR_ReleasePagelist

OCR_ReleasePagelist

OCR_ReleasePagelist releases the memory used internally by the **PXO_Pagelist** structure created by **OCR_NewPagelist**.

```
HRESULT  OCR_ReleasePagelist(
     PXO_Pagelist* pPageList
);
```

Parameters

pPageList Pointer to **PXO_Pagelist** variable previously created by **OCR_NewPagelist**.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

Example (C++)

```
PXO_Pagelist inPages;
HRESULT hr;
DWORD nPages = 23;

hr = OCR_NewPagelist(&inPages);
if (!FAILED(hr))
{
    for (DWORD i=0; i < nPages; i+=2)
    { // add all even pages up to page 23
        OCR_AddPage(inPages, i);
    }
}

// DO STUFF
// ...
//

// Release pagelist when finished
OCR_ReleasePagelist(&inPages);
```

See also

OCR_NewPagelist
OCR_AddPage
OCR_NumPages
OCR_GetPageByIndex
OCR_NewInputFields
OCR_AddInputField
OCR_NumInputFields
OCR_GetInputFieldByIndex
OCR_ReleaseInputFields
OCR_PagesToInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW
TEMPLATE FILES

Home > Input List Handling > OCR_NewInputFields

OCR_NewInputFields

OCR_NewInputFields internally initializes a new input fields list (**PXO_InputField**) structure. You can think of this structure as something like an array or `std::vector class`, but all access is via API functions. The created structure should be deleted with **OCR_ReleaseInputFields** when finished.

```
HRESULT  OCR_NewInputFields(
     PXO_InputFields *pInFields
);
```

Parameters

pInFields pointer to a variable of the type **PXO_InputField** which will receive the created page-list object.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

See also

OCR_NewPagelist
OCR_AddPage
OCR_NumPages
OCR_GetPageByIndex
OCR_ReleasePagelist
OCR_AddInputField
OCR_NumInputFields
OCR_GetInputFieldByIndex
OCR_ReleaseInputFields
OCR_PagesToInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW
TEMPLATE FILES

Home > Input List Handling > OCR_AddInputField

OCR_AddInputField

OCR_AddInputField adds a new input field (**PXO_InputField**) to the **PXO_InputFields** structure created by **OCR_NewInputFields**.

```
HRESULT  OCR_AddInputField(
     PXO_InputFields InFields,
     PXO_InputField InField
);
```

Parameters

InFields **PXO_Pagelist** variable previously created by **OCR_NewInputFields**.
InField A **PXO_InputField** structure to add to the list.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

Note

BSTR members of **PXO_InputField** will be reallocated (copied) when added to the internal **PXO_InputFields** list. The **BSTR** members of ***InField*** can thus be safely deallocated with **SysFreeString()** without impacting the internal **PXO_InputFields** list.

See also

OCR_NewPagelist
OCR_AddPage
OCR_NumPages
OCR_GetPageByIndex
OCR_ReleasePagelist
OCR_NewInputFields
OCR_NumInputFields
OCR_GetInputFieldByIndex
OCR_ReleaseInputFields
OCR_PagesToInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW
TEMPLATE FILES

Home > Input List Handling > OCR_NumInputFields

OCR_NumInputFields

OCR_NumInputFields returns the number of input page numbers currently stored in the **PXO_InputFields** structure created by **OCR_NewInputFields**.

```
HRESULT  OCR_NumInputFields(
     PXO_InputFields InFields,
     DWORD *nPages
);
```

Parameters

InFields **PXO_InputFields** variable previously created by **OCR_NewInputFields**.
nPages Pointer to a DWORD, receives the number of input pages in ***InFields***.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

See also

OCR_NewPagelist
OCR_AddPage
OCR_NumPages
OCR_GetPageByIndex
OCR_ReleasePagelist
OCR_NewInputFields
OCR_AddInputField
OCR_GetInputFieldByIndex
OCR_ReleaseInputFields
OCR_PagesToInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW
TEMPLATE FILES

Home > Input List Handling > OCR_GetInputFieldByIndex

OCR_GetInputFieldByIndex

OCR_GetInputFieldByIndex returns the 0-index specified **PXO_InputField** currently stored in the ***InFields*** structure.

```
HRESULT  OCR_GetInputFieldByIndex(
     PXO_InputFields InFields,
     DWORD nIndex,
     PXO_InputField *InField
);
```

Parameters

InFields **PXO_InputFields** variable previously created by **OCR_NewInputFields**.

nIndex Zero-based index of the field to retrieve from ***InFields***.

InField Pointer to a structure that will be filled with the requested input field. All **BSTR** members are copies of the internal data and must be deallocated with SysFreeString() (which will not impact the internally stored strings associated with the input field).

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

See also

OCR_NewPagelist
OCR_AddPage
OCR_NumPages
OCR_GetPageByIndex
OCR_ReleasePagelist
OCR_NewInputFields
OCR_AddInputField
OCR_NumInputFields
OCR_ReleaseInputFields
OCR_PagesToInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW
TEMPLATE FILES

Home > Input List Handling > OCR_ReleaseInputFields

OCR_ReleaseInputFields

OCR_ReleaseInputFields frees the memory used by an input fields list (**PXO_InputFields**) structure.

```
HRESULT  OCR_ReleaseInputFields(
     PXO_InputFields *pInFields
);
```

Parameters

pInFields pointer to a variable of the type **PXO_InputFields** which will be freed.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

Note

All internal **BSTR** members of **PXO_InputFields** are freed up by this function. All internal **BSTR**s are copies of the **BSTR**s that were originally input via an input **PXO_InputField** structure and call to **OCR_AddInputField**, so calling **OCR_ReleaseInputFields** will not impact these original input BSTR members – they must be freed separately.

See also

OCR_NewPagelist
OCR_AddPage
OCR_NumPages
OCR_GetPageByIndex
OCR_ReleasePagelist
OCR_NewInputFields
OCR_AddInputField

OCR_NumInputFields
OCR_GetInputFieldByIndex
OCR_PagesToInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW
TEMPLATE FILES

OCR_PagesToInputFields

OCR_PagesToInputFields duplicates an input field for the pages specified by an input page list structure.

```
 HRESULT  OCR_PagesToInputFields(
      PXO_InputFields InFields,
      PXO_InputField InField,
      PXO_Pagelist PageList
 );
```

Parameters

InFields **PXO_InputFields** variable previously created by **OCR_NewInputFields**.
InField Input field to be duplicated.
PageList An input **PXO_Pagelist** structure previously created by **OCR_NewPagelist** and populated with page numbers with **OCR_AddPage**.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

Notes

The BSTR members of InField are duplicated, one for each entry added to the InFields structure. As such, it is the responsibility of the developer to free up the **BSTR** members of InField with SysAllocString(), as appropriate, and conversely modifying these BSTR members will not impact the stored fields.

See also

OCR_NewPagelist
OCR_AddPage
OCR_NumPages
OCR_GetPageByIndex
OCR_ReleasePagelist
OCR_NewInputFields
OCR_AddInputField
OCR_NumInputFields
OCR_GetInputFieldByIndex
OCR_ReleaseInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW
TEMPLATE FILES

OCR_LoadTemplateW

OCR_LoadTemplateW loads a list of input fields into a **PXO_InputFields** structure from an ASCII text input file.

```
 HRESULT  OCR_LoadTemplateW(
      PXO_InputFields InputFields,
      LPWSTR lpwFilename,
 );
```

Parameters

InputFields **PXO_InputFields** variable previously created by **OCR_NewInputFields**.
lpwFilename Filename of a **Template File** to load.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

Notes

Fields read from the template file are appended to ***InputFields*** without overwriting existing data (if any). It is possible to load a template to the end of an existing list of input fields, and it is also possible to load multiple template files into a single InputFields structure with multiple calls to this function, passing the same ***InputFields*** variable in each case. Fields can be read to a blank input list, but it must have been already initialized with **OCR_NewInputFields**.

See also

- OCR_NewPagelist
- OCR_AddPage
- OCR_NumPages
- OCR_GetPageByIndex
- OCR_ReleasePagelist
- OCR_NewInputFields
- OCR_AddInputField
- OCR_NumInputFields
- OCR_GetInputFieldByIndex
- OCR_ReleaseInputFields
- OCR_PagesToInputFields
- OCR_SaveTemplateW
- TEMPLATE FILES

Home > Input List Handling > OCR_SaveTemplateW

OCR_SaveTemplateW

OCR_SaveTemplateW saves a list of input fields from a PXO_InputFields structure to a template file.

```
 HRESULT  OCR_SaveTemplateW(
     PXO_InputFields InputFields,
     LPWSTR lpwFilename,
 );
```

Parameters

InputFields PXO_InputFields variable previously created by **OCR_NewInputFields**.
lpwFilename Filename of a **Template File** to save.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

Example (C++)

```
PXO_InputFields InFields;
PXO_InputField tmpField;

OCR_NewInputFields(&InFields);
OCR_LoadTemplateW(InFields, L"z:\\sdktests\\template.pxt");
OCR_GetInputFieldByIndex(InFields, 0, &tmpField);

// Use other PDF-X OCR SDK functions
// ...

// Free input fields
OCR_ReleaseInputFields(&InFields);

// Free BSTR members copied into tmpField
SysFreeString(tmpField.blacklist);
SysFreeString(tmpField.whitelist);
SysFreeString(tmpField.label);
```

See also

- OCR_NewPagelist
- OCR_AddPage
- OCR_NumPages
- OCR_GetPageByIndex
- OCR_ReleasePagelist
- OCR_NewInputFields
- OCR_AddInputField
- OCR_NumInputFields
- OCR_GetInputFieldByIndex
- OCR_ReleaseInputFields
- OCR_PagesToInputFields
- OCR_LoadTemplateW
- TEMPLATE FILES

TEMPLATE FILES

Template files are plain text files (ASCII) containing input field information. Input checking is currently minimal, and an error will be returned if the files do not follow this format exactly. Future releases will provide more robust template input checking and perhaps an additional new template model.

Overall Structure

Variables are shown within parenthesis along with their expected type. Begin and end field text are shown in **bold.** The region's bounding box values are PDF coordinates, and can be floating point / decimal types. PDF coordinates are points (72 / inch) and the origin (0,0) is the lower left corner of the page. Positive X values are further to the right, and positive Y values are closer to the top. The format of this file is provided for your convenience; the easiest way to create template files is with the **OCR_SaveTemplateW** function.

```
(int: Total number of fields)
start_region
(int: page_number)
(float LEFT) (float BOTTOM) (float RIGHT) (float TOP)
(STRING/LINE: Blacklist)
(STRING/LINE: Whitelist)
(STRING/LINE: Label)
end_region
start_region
(...).
end_region
(...)
```

NOTES Blank string input fields must be denoted by the string NOT_DEFINED – with no additional punctuation, on a separate line.

See also

OCR_NewPagelist
OCR_AddPage
OCR_NumPages
OCR_GetPageByIndex
OCR_ReleasePagelist
OCR_NewInputFields
OCR_AddInputField
OCR_NumInputFields
OCR_GetInputFieldByIndex
OCR_ReleaseInputFields
OCR_PagesToInputFields
OCR_LoadTemplateW
OCR_SaveTemplateW

Home > High-Level Functions

High-Level Functions

This opening paragraph should describe the feature that you are documenting. Explain how it is commonly used and what the benefits are. For example: The Widget Master email link allows you to easily send information about each widget to various departments within your company. Often, the feature that you are documenting can be best explained by walking the reader through step by step. Use screenshots to illustrate the steps where possible.

1. Start the application by...
2. On the startup screen, click the...
3.

See also

PDF-X OCR SDK
Getting Started
OCR Library Types
Input List Handling
Low-Level Functions
Error Handling
Tutorials
FAQ

Home > High-Level Functions > OCR_Init

OCR_Init

OCR_Init initializes the library and sets up a new **PXODocument** into which PDFs can be loaded with **OCR_LoadW** or **OCR_LoadA**, and OCR run. After use, the object must be deleted with **OCR_Delete**.

```
HRESULT  OCR_Init(
      PXODocument *Doc,
      LPCSTR Key,
      LPCSTR DevCode
);
```

Parameters

Doc pointer to a variable of the type **PXODocument** that will receive the created PDF object.

Key pointer to a null-terminated string which contains your license key for use with. This parameter may be NULL; if so, the library will operate in 'evaluation' mode and only the first two pages of any input document can be processed by OCR functions.

DevCode pointer to a null-terminated string which contains your individual developer code. This parameter may be NULL; if so, the library will operate in 'evaluation' mode and only the first two pages of any input document can be processed by OCR functions.

Return Values

If the function succeeds, the return value is OCR_OK (0). If the function fails, the return value is **error code**.

See also

OCR_Delete
OCR_LoadW
OCR_LoadA
OCR_SaveW
OCR_SaveA
OCR_GetNumInputPages
OCR_MakeSearchable
OCR_GetText
OCR_GetField
OCR_GetFields
OCR_SetCallBack
REGIONAL OCR COMMENTS

Home > High-Level Functions > OCR_Delete

OCR_Delete

OCR_Delete releases the PDF object, created previously using the **OCR_Init** function.

You must call this function once the PDF object is no longer required or all updates are complete.

```
HRESULT  OCR_Delete(
      PXODocument *Doc
);
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init**.

Return Values

If the function succeeds, the return value is OCR_OK.

If the function fails, the return value is **error code**.

See also

OCR_Init
OCR_LoadW
OCR_LoadA
OCR_SaveW
OCR_SaveA
OCR_GetNumInputPages
OCR_MakeSearchable
OCR_GetText
OCR_GetField
OCR_GetFields
OCR_SetCallBack
REGIONAL OCR COMMENTS

Home > High-Level Functions > OCR_LoadW

OCR_LoadW

OCR_LoadW loads the specified PDF file into the specified **PXODocument** object's input layer.

```
OCR_LoadW(
PXODocument Doc,
LPWSTR pwFilename
);
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init**.

pwFilename specifies the input PDF filename.

Return Values

If the function succeeds, the return value is OCR_OK. If the function fails, the return value is **error code**.

Note: In this beta release we do not support password protected documents. A workaround is to save them as unprotected and then perform OCR. Our next release will provide support for this.

See also

OCR_Init
OCR_Delete
OCR_LoadA
OCR_SaveW
OCR_SaveA
OCR_GetNumInputPages
OCR_MakeSearchable
OCR_GetText
OCR_GetField
OCR_GetFields
OCR_SetCallBack
REGIONAL OCR COMMENTS

Home > High-Level Functions > OCR_LoadA

OCR_LoadA

OCR_LoadA loads the specified PDF file into the specified **PXODocument** object's input layer.

```
OCR_LoadA(
PXODocument Doc,
LPSTR pFilename
);
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init**.

pwFilename specifies the input PDF filename.

Return Values

If the function succeeds, the return value is OCR_OK.

If the function fails, the return value is **error code**.

See also

OCR_Init
OCR_Delete
OCR_LoadW
OCR_SaveW
OCR_SaveA
OCR_GetNumInputPages
OCR_MakeSearchable
OCR_GetText
OCR_GetField
OCR_GetFields
OCR_SetCallBack
REGIONAL OCR COMMENTS

Home > High-Level Functions > OCR_SaveW

OCR_SaveW

OCR_SaveW saves the **PXODocument** output layer to the specified file.

```
HRESULT OCR_SaveW(
  PXODocument Doc,
  LPWSTR pwFilename
);
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init** and made searchable with **OCR_MakeSearchable**.

pwFilename specifies the output PDF filename.

Return Values

If the function succeeds, the return value is OCR_OK.

If the function fails, the return value is **error code**.

See also

OCR_Init
OCR_Delete
OCR_LoadW
OCR_LoadA
OCR_SaveA
OCR_GetNumInputPages
OCR_MakeSearchable
OCR_GetText
OCR_GetField
OCR_GetFields
OCR_SetCallBack
REGIONAL OCR COMMENTS

Home > High-Level Functions > OCR_SaveA

OCR_SaveA

OCR_SaveA saves the **PXODocument** output layer to the specified file.

```
HRESULT OCR_SaveA(
  PXODocument Doc,
  LPSTR pFilename
);
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init** and made searchable with **OCR_MakeSearchable**.

pwFilename specifies the output PDF filename.

Return Values If the function succeeds, the return value is OCR_OK.

If the function fails, the return value is **error code**.

See also

OCR_Init
OCR_Delete
OCR_LoadW
OCR_LoadA
OCR_SaveW
OCR_GetNumInputPages
OCR_MakeSearchable
OCR_GetText
OCR_GetField
OCR_GetFields
OCR_SetCallBack
REGIONAL OCR COMMENTS

Home > High-Level Functions > OCR_GetNumInputPages

OCR_GetNumInputPages

OCR_GetNumInputPages returns the number of pages in the input layer loaded into the **PXODocument** by **OCR_LoadW** or **OCR_LoadA**.

```
HRESULT OCR_GetNumInputPages(
  PXODocument Doc,
  DWORD* nPages
);
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init** and loaded with one of the load functions.

nPages pointer to receive the DWORD value with the number of pages in the document’s input layer.

Return Values

If the function succeeds, the return value is OCR_OK.

If the function fails, the return value is **error code**.

See also

OCR_Init
OCR_Delete
OCR_LoadW
OCR_LoadA
OCR_SaveW
OCR_SaveA
OCR_MakeSearchable
OCR_GetText
OCR_GetField
OCR_GetFields
OCR_SetCallBack
REGIONAL OCR COMMENTS

Home > High-Level Functions > OCR_MakeSearchable

OCR_MakeSearchable

OCR_MakeSearchable processes the input document layer and generates a new output layer containing the searchable PDF results.

```
HRESULT OCR_MakeSearchable(
   PXODocument Doc,
   PXO_Options* pOptions,
   PXO_Pagelist PageList = NULL
);
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init** and loaded with one of the load functions, ie, **OCR_LoadW**.

pOptions Input pointer to a **PXO_Options** structure containing the required parameters for OCR.

PageList Optional input **PXO_Pagelist** structure containing a list of PDF pages to include in the OCR. If set to NULL (default), the function will OCR every page in the input document.

Return Values

If the function succeeds, the return value is OCR_OK.

If the function fails, the return value is **error code**.

See also

OCR_Init
OCR_Delete
OCR_LoadW
OCR_LoadA
OCR_SaveW
OCR_SaveA
OCR_GetNumInputPages
OCR_GetText
OCR_GetField
OCR_GetFields
OCR_SetCallBack
REGIONAL OCR COMMENTS

Home > High-Level Functions > OCR_GetText

OCR_GetText

OCR_GetText processes the input document layer and returns the plain text with basic text formatting (word spacing and newlines).

```
HRESULT OCR_GetText(
   PXODocument Doc,
   PXO_Options* pOptions,
   BSTR* bstrTextOut,
   PXO_Pagelist PageList = NULL,
   LPWSTR delim=L”\n”
);
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init** and loaded with one of the load functions, ie, **OCR_LoadW**.

pOptions Input pointer to a **PXO_Options** structure containing the required parameters for OCR

bstrTextOut Pointer to a BSTR variable that will receive the allocated text. Text must be deallocated with SysFreeString() when finished.

PageList Optional input **PXO_Pagelist** structure containing a list of PDF pages to include in the OCR. If set to NULL (default), the function will OCR every page in the input document.

delim Optional text delimiter to be inserted between recognized pages of text. Default is L”\n” (newline).

Return Values

If the function succeeds, the return value is OCR_OK.

If the function fails, the return value is **error code**.

See also

OCR_Init
OCR_Delete
OCR_LoadW
OCR_LoadA
OCR_SaveW
OCR_SaveA
OCR_GetNumInputPages
OCR_MakeSearchable
OCR_GetField
OCR_GetFields
OCR_SetCallBack
REGIONAL OCR COMMENTS

Home > High-Level Functions > OCR_GetField

OCR_GetField

OCR_GetField performs regional / zonal OCR on the input layer of the document, according to the rectangular region and OCR parameters specified, and returns the plain text with basic text formatting (word spacing and newlines if relevant).

```
 HRESULT OCR_GetField(
   PXODocument Doc,
   PXO_Options* pOptions,
   BSTR* bstrTextOut,
   PXO_InputField InField,
   DWORD Flags=PXO_Origin_BottomLeft
 );
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init** and loaded with one of the load functions, ie, **OCR_LoadW**.

pOptions Input pointer to a **PXO_Options** structure containing the required parameters for OCR.

bstrTextOut Pointer to a BSTR variable that will receive the allocated text. Text must be deallocated with SysFreeString() when finished.

InField Input **PXO_InputField** structure containing an input field / zone definition for regional / zonal OCR.

Flags Optional flags to specify the coordinate system used by the input fields. See **PXO_FieldInputFlags**.

Return Values

If the function succeeds, the return value is OCR_OK.

If the function fails, the return value is **error code**.

See also

OCR_Init
OCR_Delete
OCR_LoadW
OCR_LoadA
OCR_SaveW
OCR_SaveA
OCR_GetNumInputPages
OCR_MakeSearchable
OCR_GetText
OCR_GetFields
OCR_SetCallBack
REGIONAL OCR COMMENTS

Home > High-Level Functions > OCR_GetFields

OCR_GetFields

OCR_GetFields performs regional / zonal OCR on the input layer of the document, according to the rectangular regions and OCR parameters specified, and returns the plain text with basic text formatting (including word spacing and newlines if detected).

```
 HRESULT OCR_GetField(
   PXODocument Doc,
   PXO_Options* pOptions,
   BSTR* bstrTextOut,
   PXO_InputFields InFields,
   LPWSTR delim=L”\n”,
   DWORD Flags=PXO_Origin_BottomLeft
 );
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init** and loaded with one of the load functions, ie, **OCR_LoadW**.

pOptions Input pointer to a **PXO_Options** structure containing the required parameters for OCR.

bstrTextOut Pointer to a BSTR variable that will receive the allocated text. Text must be deallocated with SysFreeString() when finished.

InField Input **PXO_InputFields** structure containing an input field / zone definition for regional / zonal OCR. See also **OCR_LoadTemplateW** and **OCR_NewInputFields**.

delim Optional text delimiter to be inserted between recognized fields of text. Default is L”\n” (newline).

Flags Optional flags to specify the coordinate system used by the input fields. See **PXO_FieldInputFlags**.

Return Values

If the function succeeds, the return value is OCR_OK.

If the function fails, the return value is **error code**.

See also

OCR_Init
OCR_Delete
OCR_LoadW
OCR_LoadA
OCR_SaveW
OCR_SaveA
OCR_GetNumInputPages
OCR_MakeSearchable
OCR_GetText
OCR_GetField
OCR_SetCallBack
REGIONAL OCR COMMENTS

Home > High-Level Functions > OCR_SetCallBack

OCR_SetCallBack

OCR_SetCallBack sets the callback function to be used during the PDF rasterization process.

```
 HRESULT  PXCV_SetCallBack(
      PXODocument Doc,
      PXO_CALLBACK_FUNC clbFn,
      LPARAM clbParam
 );
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init**.

clbFn specifies the callback function, which must be defined as:

```
 typedef BOOL (__stdcall *PXO_CALLBACK_FUNC) (DWORD dwStage, DWORD dwLevel, LPARAM param);
```

The first parameter of this function indicates the callback state; the second indicates the progress level (see table below), and the third will always have the same value as passed in *UserData*.

Note: The maximum level will always be the number of input pages (or fields) times four. Thus (dwLevel - 1) / 4 (rounded down) gives the current working page number, and the remainder (dwlevel % 4) gives the stage for that page (1=Rasterizing, 2=Auto-rotating and pre-processing image, 3=running OCR, 0 = Preparing output).

Callback function's state constants table

CONSTANT	VALUE	MEANING OF dwLevel
`PXOClb_Start`	1	`MaxVal` - maximum value of the level which will be passed
`PXOClb_Processing`	2	Current progress level - any value from `0` to `MaxVal`
`PXOClb_Finish`	3	May be any value from `0` to `MaxVal` (`MaxVal` if all passed), may be ignored

Note: The Callback function should return TRUE (any non-zero value) to continue processing or FALSE (zero) to abort the operation.

UserData specifies a user-defined callback parameter to be passed as a third parameter to the function specified by *pProc*.

Return Values

If the function succeeds, the return value is DS_OK.

If the function fails, the return value is **error code**.

See also

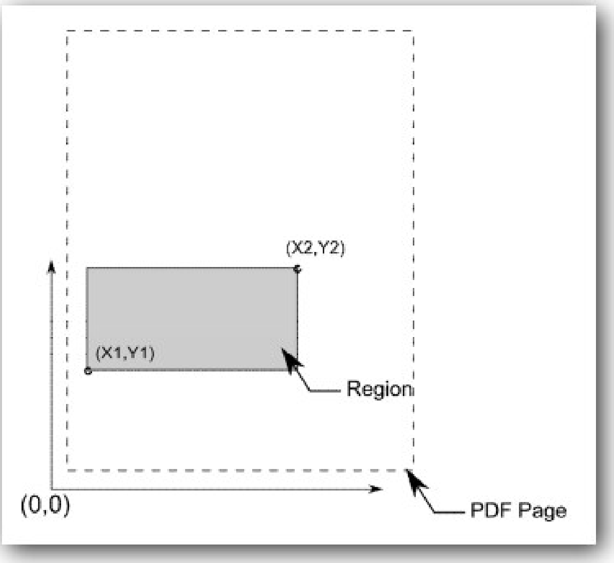
OCR_Init
OCR_Delete
OCR_LoadW
OCR_LoadA
OCR_SaveW
OCR_SaveA
OCR_GetNumInputPages
OCR_MakeSearchable
OCR_GetText
OCR_GetField
OCR_GetFields
REGIONAL OCR COMMENTS

Home > High-Level Functions > REGIONAL OCR COMMENTS

REGIONAL OCR COMMENTS

REGIONAL OCR COMMENTS

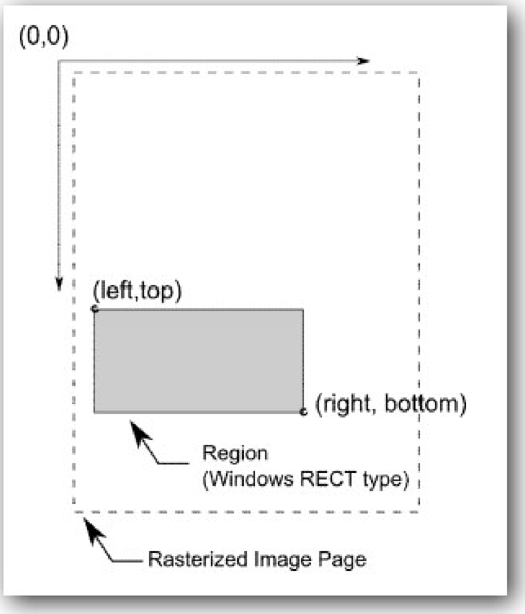
OCR Fields are defined in PDF coordinates which begin at (0,0) at the bottom left corner of the page by default, but as of version 1.0.5 may also be defined relative to the top corner of the page (see **PXO_FieldInputFlags** for details). The **PXO_InputField** structure contains the geometric definitions for this region, along with OCR parameters such as the whitelist and blacklist. Coordinates are in points (72.0 per inch), and the fields X1,Y1 and X2,Y2 form the coordinate points shown in the figure to the right.



A number of default whitelist and blacklist character strings are provides in the **ocrdefaults.h** header file.

The PXO_Options field ImageFlags has special meaning for the regional OCR performed by **OCR_GetField** and **OCR_GetFields**. Rather than specifying that the whole page be rotated, this rotates the sub-image specified by the regional rectangle. This is to avoid situations where rotation of the page would result in the regional boundaries changing, but still allows the increased OCR accuracy given by automatic image rotation.

The low level functions **OCRp_Field** and **OCRp_Page** return structures which give access to **OCR_SymbolBox** structures. All coordinates provided in these structures refer to Raster Page Coordinates (calculated from the internal image used for OCR of the page). These coordinates differ from PDF Page coordinates, as shown to the right. The origin (0,0) is in the upper left hand corner of the page, and increasing Y-axis values refer to lower positions on the page, similar to the coordinate system used in Windows Device Contexts and RECT structures.



The **OCR_RasterPageSettings** structure contains a member, ***scalefactor***, which can be used to convert coordinates, taking into account the origin difference. **OCRp_RasterRectToPDF** is provided as a convenience to convert Windows RECT structures to PDF coordinates.

See also

OCR_Init
OCR_Delete
OCR_LoadW
OCR_LoadA
OCR_SaveW
OCR_SaveA
OCR_GetNumInputPages
OCR_MakeSearchable
OCR_GetText
OCR_GetField
OCR_GetFields
OCR_SetCallBack

Home > Low-Level Functions

Low-Level Functions

This opening paragraph should describe the feature that you are documenting. Explain how it is commonly used and what the benefits are. For example: The Widget Master email link allows you to easily send information about each widget to various departments within your company. Often, the feature that you are documenting can be best explained by walking the reader through step by step. Use screenshots to illustrate the steps where possible.

1. Start the application by...
2. On the startup screen, click the...
3.

See also

PDF-X OCR SDK
Getting Started
OCR Library Types
Input List Handling
High-Level Functions
Error Handling
Tutorials
FAQ

Home > Low-Level Functions > OCRp_Page

OCRp_Page

OCRp_Page performs regional / zonal OCR on the input layer of the document and specified page, and returns the results in a structure that can be queried for text layout details.

```
HRESULT OCRp_Page(
   PXODocument Doc,
   ULONG nPage,
   PXO_Options* pOptions,
   PXO_Page *pPageResults,
   OCR_RasterPageSettings *pRasterSettings,
);
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init** and loaded with one of the load functions, ie, **OCR_LoadW**.

nPage Input PDF page number (zero-indexed).

pOptions Input pointer to a **PXO_Options** structure containing the required parameters for OCR.

pPageresults Pointer to a **PXO_Page** variable that will receive the results. Once finished, this structure must be freed with **OCRp_FreePage**.

pRasterSettings Pointer to an **OCR_RasterPageSettings** variable that will receive the rasterization settings for the page image (needed to convert coordinates from page formatting information to PDF coordinates).

Return Values

If the function succeeds, the return value is DS_OK.

If the function fails, the return value is **error code**.

See also

OCRp_Field
OCRp_PageText
OCRp_RegionCountFromPage
OCRp_GetRegionFromPage
OCRp_SymbolCountFromRegion
OCRp_GetSymbolFromRegion
OCRp_RasterRectToPDF
OCRp_FreePage

OCRp_Field

OCRp_Field performs regional / zonal OCR on the input layer of the document, according to the rectangular regions and OCR parameters specified, and returns the results in a structure that can be queried for text layout details.

```
HRESULT OCRp_Field(
  PXODocument Doc,
  PXO_InputField InField,
  PXO_Options* pOptions,
  PXO_Page *pPageResults,
  OCR_RasterPageSettings *pRasterSettings,
);
```

Parameters

Doc specifies the PDF object previously created by the function **OCR_Init** and loaded with one of the load functions, ie, **OCR_LoadW**.

InField Input **PXO_InputField** structure.

pOptions Input pointer to a **PXO_Options** structure containing the required parameters for OCR.

pPageresults Pointer to a **PXO_Page** variable that will receive the results. Once finished, this structure must be freed with **OCRp_FreePage**.

pRasterSettings Pointer to an **OCR_RasterPageSettings** variable that will receive the rasterization settings for the page image (needed to convert coordinates from page formatting information to PDF coordinates).

Return Values

If the function succeeds, the return value is OCR_OK.

If the function fails, the return value is **error code**.

See also

OCRp_Page
OCRp_PageText
OCRp_RegionCountFromPage
OCRp_GetRegionFromPage
OCRp_SymbolCountFromRegion
OCRp_GetSymbolFromRegion
OCRp_RasterRectToPDF
OCRp_FreePage

OCRp_PageText

OCRp_PageText returns plain text from the specified **PXO_Page** structure, with minimal text formatting (words and newlines).

```
HRESULT OCRp_PageText(
  PXO_Page PageResults,
  BSTR* bstrTextOut
);
```

Parameters

PageResults Specifies the **PXO_Page** object containing results to convert to minimally formatted text.

bstrTextOut Pointer to a new BSTR variable to contain the text output, allocated by the function. Must be freed when finished with **SysFreeString()**.

Return Values

If the function succeeds, the return value is DS_OK.

If the function fails, the return value is **error code**.

See also

OCRp_Page
OCRp_Field
OCRp_RegionCountFromPage
OCRp_GetRegionFromPage
OCRp_SymbolCountFromRegion

OCRp_GetSymbolFromRegion
OCRp_RasterRectToPDF
OCRp_FreePage

Home > Low-Level Functions > OCRp_RegionCountFromPage

OCRp_RegionCountFromPage

OCRp_RegionCountFromPage returns number of regions in the specified page.

```
HRESULT OCRp_RegionCountFromPage(
  PXO_Page PageResults,
  ULONG* pRegionCount
);
```

Parameters

PageResults Specifies the **PXO_Page** object containing results to retrieve the region count from.

pRegionCount Pointer to a ULONG (unsigned long) variable to contain the region count.

Return Values

If the function succeeds, the return value is DS_OK.

If the function fails, the return value is **error code**.

See also

OCRp_Page
OCRp_Field
OCRp_PageText
OCRp_GetRegionFromPage
OCRp_SymbolCountFromRegion
OCRp_GetSymbolFromRegion
OCRp_RasterRectToPDF
OCRp_FreePage

Home > Low-Level Functions > OCRp_GetRegionFromPage

OCRp_GetRegionFromPage

OCRp_GetRegionFromPage returns the requested output region from page.

```
HRESULT OCRp_GetRegionFromPage(
  PXO_Page PageResults,
  ULONG nRegion,
  PXO_Region *pRegionResults
);
```

Parameters

PageResults Specifies the **PXO_Page** object containing results to retrieve a region from.

nRegion Index (zero-based) specifying the region number to retrieve. Maximum value is the region count returned by **OCRp_RegionCountFromPage** minus one.

pRegionResults Pointer to a **PXO_Region** variable which will contain the new region results.

NOTE: Regions are internally linked to their parent PXO_Page. A call to OCRp_FreePage on the parent region will free the memory used by, and invalidate, any region retrieved by this function.

Return Values

If the function succeeds, the return value is DS_OK.

If the function fails, the return value is **error code**.

See also

OCRp_Page
OCRp_Field

OCRp_PageText
OCRp_RegionCountFromPage
OCRp_SymbolCountFromRegion
OCRp_GetSymbolFromRegion
OCRp_RasterRectToPDF
OCRp_FreePage

Home > Low-Level Functions > OCRp_SymbolCountFromRegion

OCRp_SymbolCountFromRegion

OCRp_SymbolCountFromRegion returns number of symbols in the specified region.

```
HRESULT OCRp_SymbolCountFromRegion(
  PXO_Region RegionResults,
  ULONG* pSymbolCount
);
```

Parameters

RegionResults Specifies the **PXO_Region** object containing results to retrieve the symbol count from.

pRegionCount Pointer to a ULONG (unsigned long) variable to contain the symbol count.

Return Values

If the function succeeds, the return value is DS_OK.

If the function fails, the return value is **error code**.

See also

OCRp_Page
OCRp_Field
OCRp_PageText
OCRp_RegionCountFromPage
OCRp_GetRegionFromPage
OCRp_GetSymbolFromRegion
OCRp_RasterRectToPDF
OCRp_FreePage

Home > Low-Level Functions > OCRp_GetSymbolFromRegion

OCRp_GetSymbolFromRegion

OCRp_GetSymbolFromRegion returns requested symbol from output region.

```
HRESULT OCRp_GetSymbolFromRegion(
  PXO_Region RegionResults,
  DWORD nSymbol,
  OCR_SymbolBox *pSymbolBox
);
```

Parameters

RegionResults Specifies the **PXO_Region** object containing results to retrieve a symbol from.

nSymbol Index (zero-based) specifying the symbol number to retrieve. Maximum value is the symbol count returned by **OCRp_SymbolCountFromRegion** minus one.

pSymbolBox Pointer to a **OCR_SymbolBox** variable which will contain the new region results.

NOTE: Symbols are copied into the variable specified by the pSymbolBox pointer. A call to OCR_FreePage does not destroy their data.

Return Values

If the function succeeds, the return value is DS_OK.

If the function fails, the return value is **error code**.

See also

OCRp_Page
OCRp_Field
OCRp_PageText
OCRp_RegionCountFromPage
OCRp_GetRegionFromPage
OCRp_SymbolCountFromRegion
OCRp_RasterRectToPDF
OCRp_FreePage

Home > Low-Level Functions > OCRp_RasterRectToPDF

OCRp_RasterRectToPDF

OCRp_RasterRectToPDF returns the coordinates of the input raster page rectangle in PDF coordinates.

```
HRESULT OCRp_RasterRectToPDF(
  RECT SourceRect,
  OCR_RasterPageSettings RasterSettings,
  double *left,
  double *bottom,
  double *right,
  double *top
);
```

Parameters

SourceRect Specifies input **RECT** coordinates in rasterized image page coordinates (Windows origin; 0,0 at upper left, coordinates in pixels).

RasterSettings Raster settings returned from **OCRp_Page** or **OCRp_Field** to use to convert to PDF coordinates

left,

bottom,

right,

top Pointers to double variables which receive the equivalent PDF-coordinates rectangle (origin 0,0 at lower left, coordinates in printers’ points).

NOTE: See Regional OCR Comments for more explanation.

Return Values

If the function succeeds, the return value is DS_OK.

If the function fails, the return value is **error code**.

See also

OCRp_Page
OCRp_Field
OCRp_PageText
OCRp_RegionCountFromPage
OCRp_GetRegionFromPage
OCRp_SymbolCountFromRegion
OCRp_GetSymbolFromRegion
OCRp_FreePage

Home > Low-Level Functions > OCRp_FreePage

OCRp_FreePage

OCRp_FreePage returns number of regions in the specified page.

```
HRESULT OCRp_FreePage(
  PXO_Page *pPageResults
);
```

Parameters

pPageResults A pointer to the **PXO_Page** object to be destroyed.

Return Values

If the function succeeds, the return value is DS_OK.

If the function fails, the return value is **error code**.

NOTE: This function destroys the PXO_Page object and all of its associated PXO_Region objects. Any PXO_Region variables (actually void* pointers to internal data structures) retrieved by calls to OCRp_GetRegionFromPage will be invalid after calling this function on the parent PXO_Page.

See also

OCRp_Page
OCRp_Field
OCRp_PageText
OCRp_RegionCountFromPage
OCRp_GetRegionFromPage
OCRp_SymbolCountFromRegion
OCRp_GetSymbolFromRegion
OCRp_RasterRectToPDF

Home > Error Handling

Error Handling

This opening paragraph should describe the feature that you are documenting. Explain how it is commonly used and what the benefits are. For example: The Widget Master email link allows you to easily send information about each widget to various departments within your company. Often, the feature that you are documenting can be best explained by walking the reader through step by step. Use screenshots to illustrate the steps where possible.

1. Start the application by...
2. On the startup screen, click the...
3.

See also

PDF-X OCR SDK
Getting Started
OCR Library Types
Input List Handling
High-Level Functions
Low-Level Functions
Tutorials
FAQ

Home > Error Handling > Error Codes

Error Codes

Most functions return an HRESULT value which provides a simple means to determine the success or otherwise of a function call.

If the most significant bit or result is set to 1 then the specified error occurred, otherwise the function was successful. Here are two simple macros for C/C++ which apply these checks:

```
#define IS_DS_SUCCESSFUL(x)  (((x) & 0x80000000) == 0)
#define IS_DS_FAILED(x)      (((x) & 0x80000000) != 0)
```

Note: It is strongly recommended to always use the specified (or equivalent macro's) to establish if the function call was successful or otherwise. A simple comparison with 0 (zero) will usually provide erroneous and unreliable results described in the following example scenario's.

A Function may return a warning with a code that is not equal to zero (and also not negative!). This usually means that the function has succeeded and is providing additional information about the call. i.e. The function did not find any OCR text and is returning an empty result. For more information see the description provided for each particular error code and function.

To determine if the return value is generating a warning we provide the **IS_DS_WARNING** macros.

This would be the correct syntax to check for the error status of the **PXCV_CheckPassword** function:

```
HRESULT hr = PXCV_CheckPassword(doc, password, len);
 if (IS_DS_FAILED(hr))
 {
  // An error occurred!
  // Manage the error accordingly to provide an orderly exit from the function call
  ...
 }
 else
 {
  // 'hr' contains value which indicate which password was supplied - owner or user
  ...
 }
```

Note: The example code below demonstrates how NOT to provide error checking in your code

```
HRESULT hr = PXCV_CheckPassword(doc, password, len);
```

```
 if (hr == 0)
 {
  // treat as success
  ...
  (this is not true as a positive return value was received!)
  ...
 }
 else
 {
  // treat as error
  (Incorrect as the return value has not been adequately identified and this is
unreliable!)
  ...
 }
```

Most frequently returned error codes are listed in the following table, however functions may return additional codes which are not listed here.

Possible values of errors from the PDF-X OCR SDK:

CONSTANT	VALUE	DESCRIPTION
OCR_ERR_NOTIMPL	0x820a04b0	Not implemented
OCR_ERR_INVARG	0x820a0001	Invalid argument
OCR_ERR_MEMALLOC	0xc20a03e8	Insufficient memory
OCR_ERR_USER_BREAK	0xc20a01f4	Operation aborted by user
OCR_WRN_USER_BREAK	0x420a2710	Operation completed but user requested break
OCR_ERR_DOCNOTREAD	0x820a2711	Input document empty
OCR_ERR_WRONGPAGENUMBER	0x820a2712	Invalid page requested
OCR_ERR_DOCHASNOPAGES	0x820a2713	Document has no pages
OCR_ERR_NOTLICENSED	0x820a2714	Operation requested not allowed by license
OCR_WRN_NOTLICENSED	0x420a2715	Operation completed but some portions not allowed
OCR_ERR_INTERNAL	0x820a2717	Unspecified internal error
OCR_ERR_POINTER	0x820a2718	NULL pointer received
OCR_WRN_NORESULTS	0x420a2716	Operation completed but no results were found
OCR_WRN_NOTROTATED	0x420a2719	Operation completed but one or more pages were not auto rotated due to failure determining rotation angle (possibly because page blank)
OCR_WRN_FIELDBOUNDS	0x420a271a	Operation completed but one or more **PXO_InputField** structures specified rectangles that were partially outside of the bounds of the page. Rectangles were truncated to fit PDF page.
OCR_ERR_FIELDBOUNDS	0x820a271b	Operation did not complete because one or more **PXO_InputField** structures specified rectangle(s) that were either empty or wholly outside of the PDF page boundaries.

Comments

Additional error codes originating from internal PDF manipulation functions and windows API functions may be returned. There is an additional utility included with this library which provides valuable data regarding all known error codes - **DSErrorLookUp.exe**. This can be found in your PDF-XChange/Tools installation folders and is extremely useful during your application development process - we strongly recommend ALL developers utilize **DSErrorLookUp.exe** during the debugging of their applications and prior to support requests relating to Error Code return values and their meaning. Please note that OCR error codes may not be immediately available in the **DSErrorLookUp** utility during this initial beta period.

See also

Home > Tutorials

Tutorials

Articles in this section

Tutorial 1 Tutorial 2

See also

PDF-X OCR SDK
Getting Started
OCR Library Types
Input List Handling
High-Level Functions
Low-Level Functions
Error Handling
FAQ

Home > Tutorials > Tutorial 1

Tutorial 1

This opening paragraph should describe the feature that you are documenting. Explain how it is commonly used and what the benefits are. For example: The Widget Master email link allows you to easily send information about each widget to various departments within your company. Often, the feature that you are documenting can be best explained by walking the reader through step by step. Use screenshots to illustrate the steps where possible.

1. Start the application by...
2. On the startup screen, click the...
3.

See also

Tutorial 2

Home > Tutorials > Tutorial 2

Tutorial 2

This opening paragraph should describe the feature that you are documenting. Explain how it is commonly used and what the benefits are. For example: The Widget Master email link allows you to easily send information about each widget to various departments within your company. Often, the feature that you are documenting can be best explained by walking the reader through step by step. Use screenshots to illustrate the steps where possible.

1. Start the application by...
2. On the startup screen, click the...
3.

See also

Tutorial 1

Articles in this section

General FAQ

Installation FAQ

Home > FAQ > General FAQ

General FAQ

This opening paragraph should describe the feature that you are documenting. Explain how it is commonly used and what the benefits are. For example: The Widget Master email link allows you to easily send information about each widget to various departments within your company. Often, the feature that you are documenting can be best explained by walking the reader through step by step. Use screenshots to illustrate the steps where possible.

1. Start the application by...
2. On the startup screen, click the...
3.

See also

Installation FAQ

Home > FAQ > Installation FAQ

Installation FAQ

This opening paragraph should describe the feature that you are documenting. Explain how it is commonly used and what the benefits are. For example: The Widget Master email link allows you to easily send information about each widget to various departments within your company. Often, the feature that you are documenting can be best explained by walking the reader through step by step. Use screenshots to illustrate the steps where possible.

1. Start the application by...
2. On the startup screen, click the...
3.

See also

General FAQ

Index