

Foxit Reader ActiveX SDK 2.0

开发手册

©2008 福昕软件有限公司

版权所有

2008.2

目录

终述	3
设计指南	4
参考	5
属性	5
方法	7
1)打开和关闭 PDF 文档.....	7
2) 查找.....	8
3) 书签	10
4)页面布局	10
5) 保存.....	11
6)其它	11
事件	19

终述

Foxit Reader SDK ActiveX 是一个可以显示 PDF 文档的可视组件,它可以方便的集成到大多数的应用程序中去。它是由福昕软件公司为 PDF 文档应用而开发完成的,具有资源需求低以及体积小的特点.

Foxit Reader SDK ActiveX 采用了与 Foxit Reader 相同的渲染引擎.因此,能被 Foxit Reader 打开的文档也同样能够被 ActiveX 所打开,且具有同样的高质量与高速度.

Foxit Reader SDK 的 ActiveX 版本使用起来比 DLL 版本更容易,且它还具备了丰富的特性。开发者可以将该组件拖放到应用程序当中,为应用程序添加显示 PDF 文件的功能。ActiveX 可以允许用户导航, 缩放, 旋转, 滚动页面和打印所有的 PDF 文档。

2.0 版包含了更加高级的特性。它支持标注, 并且允许用户填充、导入与导出 PDF 表单。2.0 版包含了更多的导出函数和事件, 给予了程序员更加灵活的组件操控方式及对 PDF 文档更多的访问。

Foxit 提供了两个版本的 ActiveX 2.0。一个是标准版。它不包含以下特性: 创建/编辑标注, 导入/导出表单数据, 运行 javascript, 转换 PDF 文档到文本文件等等。另外一个专业版, 它包含了这些特性。您可根据应用程序的需要来选择合适的版本。对于标准版, 您可在线购买它。如果是专业版, 您可以联系 sales@foxitsoftware.com 来进一步探讨许可的细节。在开发手册当中, 我们将所有只能在专业版当中使用的属性和方法都用一个*号来表明。

Foxit Reader SDK ActiveX 可以运行在 Windows 95/NT 或更新的操作系统版本上。它是个完全独立的组件, 不需要安装其它的 PDF 软件, 比如福昕阅读器。为了能成功在 Windows 环境下注册 ActiveX 组件, 用户可能需要管理员的权限。

有不少的运用了不同的编程语言的例子程序, 包括 Visual Basic, Visual C++, Delphi 等展示了怎样去使用 ActiveX 的各种属性和方法。您可以从网址 www.foxitsoftware.com 去下载它们。

这个只是 SDK ActiveX, 并不是 Browser ActiveX。它可以为开发人员所使用, 但不能被终端用户直接使用。福昕软件将开发一个独立的 Browser ActiveX, 并

在完成的时候发布。

解锁码：如果您购买了 Foxit Reader SDK ActiveX 并且也收到了全功能版的 ActiveX 和解锁码，您可以在调用 ActiveX 当中的任何接口函数之前先调用 UnLockActiveX 这个函数来进行解锁(这个函数在‘参考’部分有描述)。如果您仅需要试用 ActiveX，那么就不需要调用这个函数

设计指南

Foxit Reader SDK ActiveX 控件是以单个文件的形式提供的，可以使用”regsvr32 foxitreader_ax.ocx”命令来安装它。如果 foxitreader_ax.ocx 不是存放在当前的目录中，您可能需要指定正确的路径。ActiveX 控件为您处理了用户接口。它也提供了一些属性和方法可以让你控制 ActiveX 的行为。在以下的例子当中，假设我们已经安装了名为 FoxitReaderSDK 的 ActiveX 控件。

指南 1：打开一个 PDF 文档。

我们将打开一个名为”testdoc.pdf”的 PDF 文档，并跳转到文档的第三页。
//注意：如果您在试用 ActiveX，那么您不需要为它解锁，但试用的标志也会
//显示在您所有文档页面的右上角。如果不想这样，那么付费的用户可以先为
//ActiveX 解锁。

```
FoxitReaderSDK.UnLockActiveX(“license_id”,“unlock_code”);  
FoxitReaderSDK.OpenFile(“testdoc.pdf”,“”);
```

指南 2：跳转到指定的页面。

我们将跳转到文档的第三页。
FoxitReaderSDK.GotoPage(2). //第一页的索引为 0

指南 3：缩放页面。

如果你想以页面的初始大小显示，可以使用以下的代码。

```
FoxitReaderSDK.SetZoomLevel(0)  
或  
FoxitReaderSDK.SetZoomLevel(100);
```

如果你想要显示文档页面初始大小的两倍大小，你可以使用以下的代码：

```
FoxitReaderSDK.SetZoomLevel(200);
```

更多的信息，请参考本设计手册的”参考”部分。

指南 4：旋转页面。

PDF 页面能够以四个方向来显示，垂直，旋转 90 度，旋转 180 度，或旋转 270 度。您只需要调用 SetRotate 这个函数就可以实现不同方向的旋转显示。

如果您想要顺时针旋转页面 90 度，则可在 VC 中使用以下的代码：

```
FoxitReaderSDK.SetRotate (1);
```

指南 5: 打印。

您所需要的仅是调用 `PrintWithDialog` 这个方法, 然后打印对话框便会弹出来, 用户可以指定打印参数并打印文档。如果您想在不弹出打印对话框的情况下打印文档, 您可以使用 `IPDFPrinter` 接口。

指南 6: 隐藏或显示用户界面元素。

您可以调用 `ShowToolBar` 来显示或隐藏工具条, 同样, 您可以调用 `ShowBookmark` 来显示隐藏书签和调用 `ShowStatusBar` 来显示或隐藏状态条。如果您想创建您自己的工具条, 您可以隐藏 `ActiveX` 内置的工具条, 再在 `ActiveX` 之外创建自己的工具条。

指南 7: 读取书签树。

您可以调用 `GetOutlineFirstChild`, `GetOutlineNextSibling` 来递归读取整个书签的树形结构。您可以从 `IPDFOutline` 这个接口来获取每个书签的信息。

指南 8: 查找文档

你可以调用 `FindFirst` 来查找所要查找的文本在整个文档当中的第一个匹配项, 当没有匹配的文本被找到, 返回值为 0。如果有匹配项, 则返回值为非 0, 并且找到的文本也会被高亮显示。然后再调用 `FindNext` 查找出在这个文档当中下一个匹配项。

如果您想在不打开文件的情况下进行查找, 您可以调用 `FindFileFirst` 和 `FindFileNext`。

指南 9: 注释

终端用户可以使用标注工具来绘制线条, 圆, 以及其它的形状来给文档添加注释。如果你想为 PDF 文档插入直线标注, 你可以设置属性 `CurrentTool` 的值为 "Line Tool", 然后你就可以在文档当中画线了。

参考

这部分描述了 `ActiveX` 提供的所有属性与方法。请注意所有的示例代码均以 C 语法来演示。如果您以 C/C++ 以外的语言进行编程, 则需要做必要的变更。

注意: 标明为 (*) 的函数只能对专业版适用。

属性

`FilePath` `BSTR`, read-only:

打开的 PDF 文件的绝对路径, 如果没有文件打开, 那么就为空值。

`PageCount` `long`, read-only:

当前打开的 PDF 文档的总页数。

`CurPage` `long`, read-only:

当前 PDF 文档的页面索引, 页面索引从 0 表示的第一页开始。

`Rotate` `short`, read and write:

当前的的旋转方向。值可以为:

- 0(正常显示);
- 1(顺时针旋转 90 度);
- 2 (旋转 180 度);
- 3 (逆时针旋转 90 度).

Zoomlever long,read and write

正常情况下 ,这个值为 10 到 1600 之间。指定特殊的值可以达到特殊的效果 ,
如下 :

- 0= 显示页面的真实大小 ,它与缩放因子被设为 100%的效果一致。
- 1= 缩放 PDF 页面到合适比例 ,使整个页面能够在客户区窗口当中全部显示。
- 2= 缩放 PDF 页面到合适比例 ,使页面的宽度能够与客户区窗口宽度相符合。

CurrentTool BSTR, read and write :

读取与设置当前的工具 ,其值可以为 :

- “ Hand Tool”
- “ ZoomOut Tool”
- “ ZoomIn Tool”
- “ TypeWriter Tool ” (*)
- “ Select Text Tool”
- “ Find Text Tool”
- “ Snapshot Tool”
- “ Annot Tool ” (*)
- “ Rectangle Link Tool ” (*)
- “ Quadrilateral Link Tool ” (*)
- “ Arrow Tool ” (*)
- “ Line Tool ” (*)
- “ Dimension Tool ” (*)
- “ Square Tool ” (*)
- “ Rectangle Tool ” (*)
- “ Circle Tool ” (*)
- “ Ellipse Tool ” (*)
- “ Polygon Tool ” (*)
- “ Cloudy Tool ” (*)
- “ Polyline Tool ” (*)
- “ Pencil Tool ” (*)
- “ Rubber Tool ” (*)
- “ Highlight Tool ” (*)
- “ Underline Tool ” (*)
- “ Strikeout Tool ” (*)
- “ Squiggly Tool ” (*)
- “ Caret Tool ” (*)
- “ Note Tool ” (*)
- “ Push Button Tool ” (*)
- “ Check Box Tool ” (*)
- “ Radio Button Tool ” (*)
- “ Combo Box Tool ” (*)

“ List Box Tool ” (*)
“ Text Field Tool ” (*)
“ Distance Tool ” (*)
“ Perimeter Tool ” (*)
“ Area Tool ” (*)
“ Image Tool ” (*)
“ FileAttachment Tool ” (*)
“ Attach a file ” (*)

等等。

您可调用 CountTools 获取当前可用的工具总数，并调用 GetToolByIndex 获取工具名称。

Printer IPDFPrinter, read-only

Printer 属性可以返回一个 IPDFPrinter 接口，您可以使用这个接口来设置打印选项并打印。

bHasFormFields (*) BOOL, read-only

如果 bHasFormFields 为真，则文档中包含有表单。

DocumentInfo (*) IPDFDocumentInfo*, read-only:

返回一个 IPDFDocumentInfo 接口，您可以利用这个接口来获取像作者，创建者，创建日期，关键字，修改日期，制作者，主题，标题等信息。

bHighlightFormFields (*) BOOL, read and write

如果 bHighlightFormFields 为真，则表单域将被高亮。

FormFieldsHighlightAlpha (*) short, read and write

表示 256 级的表单高亮颜色的透明度。

FormFieldsHighlightColor (*) OLE_COLOR, read and write

表示表单的高亮颜色。

方法

1) 打开和关闭 PDF 文档

函数：OpenFile

从本地磁盘或 HTTP 服务器上打开一个 PDF 文件。

注意：用这个方法打开的文件不会被锁定，其它的应用程序也可以打开它。

原型：

BOOL OpenFile (BSTR file_path, long password)

参数：

file_path - PDF 文件路径（包括文件扩展名），可以是 HTTP 服务器上的文件。

password - 字符串做为 PDF 文件的密码，如果不需要密码，则字符串值可以为空。

返回值：

PDF 文件被成功打开则返回非 0，否则返回 0。

函数：OpenMemFile

打开一个保存在内存当中的 PDF 文件。

原型：

BOOL OpenMemFile(long pBuffer, long size, BSTR password)

参数：

- pBuffer -调用者提供的包含 PDF 数据的缓冲区指针。
- size -由 pBuffer 所指的缓冲区的大小。
- password - PDF 文件密码字符串,如果不需要密码,则置空值。

返回值：

如果 PDF 文件成功打开则返回非 0,否则返回 0.

函数：OpenStream

从 IStream 接口打开一个 PDF 文件

原型

BOOL OpenStream (IStream* Stream, BSTR password)

参数：

- stream - IStream 接口
- password - PDF 文件密码字符串,如果不需要密码,则置空值

返回值

如果成功打开则返回非 0,否则为 0.

函数：OpenCustomFile

从一个自定义的文件访问机制里打开一个 PDF 文件,当您的程序调用这个方法时,ActiveX 将会触发两个事件 CustomFileGetSize 和 CustomFileGetBlock. 在内部的事件处理中,您的程序将以自定义的格式打开一个 PDF 文档,并返回文件的大小和数据块。更多的信息可以参考 CustomFileGetSize 和 CustomFileGetBlock 这两个函数的描述。

原型:

BOOL OpenCustomFile(BSTR password)

参数：

password - PDF 文件密码字符串,如果不需要密码,则置空值

返回值：

如果 PDF 文件成功打开,则返回非 0,否则返回 0.

CloseFile

关闭当前加载的文档。

2) 查找

FindFirst

在文档当中查找字符串,如果函数找到了字符串的第一个出现的位置,则将会跳转到所处的页面,更新 CurPage 这个属性,高亮找到的文本并返回 true,否则将返回 false.

原型

BOOL FindFirst(BSTR searchstring, BOOL bMatchCase, BOOL bMatchWholeWord)

参数：

- Searchstring -所要查找的字符串
- bmatheCase -是否大小写敏感
- bMatchWoleWord -是否全字匹配

返回值：

如果找到字符串,则返回非 0,否则返回 0.

FindNext

查找给定文本在当前文档中的下一个出现的位置。如果 ActiveX 找到了下一个出现的位置，则将会跳转到所处的页面，更新 CurPage 这个属性，高亮找到的文本并且返回 true。否则将返回 false。请注意，FindNext 将使用与 FindFirst 相同的搜索条件，包括 bMatchCase, bMatchWholeWord。如果 bSearchDown 为 true，则将向下搜索文档，如果为 false，则将向上搜索文档。

原型：

```
BOOL FindNext(BOOL bSearchDown);
```

参数：

bSearchDown -搜索方向

返回值：

如果没有匹配的文字被找到，返回值为 0，否则为非 0。

FindFileFirst.

查找指定的文件当中的字符串，如果找到字符串，则返回一个 IFindResult 接口，否则返回 null。该方法允许您在不打开 PDF 文件的情况下进行查找。比如，您可能想要对一个文件夹当中的所有的 PDF 文档来查找某个关键字，就可以遍历文件夹当中的所有 PDF 文件，并一个个的用关键字来查找。如果 ActiveX 找到了一个出现的位置，它将返回一个包含所有有关出现位置信息的 IFindResult 接口。然后您可以调用 GotoPageView 这个函数来打开文件并跳转到目标页面，高亮所出现的文本。

原型：

```
IFindResult * FindFileFirst(BSTR filename , BSTR searchstring, BOOL  
bMatchCase, BOOL bMatchWholeWord)
```

参数：

fileName - PDF 文件路径
searchstring -所要查找的字符串
bmatchcase -是否大小写敏感
bMatchWholeWord -是否全字匹配

返回值：

如果匹配字符被找到，则返回一个 IFindResult 接口，否则返回 null。

函数：FindFileNext

在全文当中查找下一个出现的字符串。

原型：

```
IFindResult * FindFileNext ();
```

参数：

无

返回值：

如果没有匹配的字符串被找到，则返回值为空，否则为 IFindResult 接口。

GotoPageView

显示和高亮所查找的结果。

Prototype:

```
void GotoPageView(IFindResult* findresult);
```

参数：

findresult -由 FindFileFirst 和 FindFileNext 所返回的 IFindResult 接口。

返回值：

空

SearchAndHighlightAllTextOnPage

查找并高亮给定的关键字在指定页面上的所有实例。

原型:

```
void SearchAndHighlightAllTextOnPage(BSTR searchstring, boolean bMatchCase, boolean bMatchWholeWord, long pageNo);
```

参数:

PageNo - 所要查找的页面的索引。

返回值

无

3) 书签

GetOutlineFirstChild

取得当前书签节点的第一个孩子节点。

原型：

```
IPDFOutline* GetOutlineFirstChild( IPDFOutline* outline)
```

参数：

outline -此书签节点将返回第一个（如果存在的话）孩子节点。如果值为 NULL,则将返回书签树的第一个节点

返回值：

如果当前节点有孩子节点，则返回其第一个孩子节点，否则返回空。

GetOutlineNextSibling

取得下一个兄弟节点。

原型：

```
IPDFOutline* GetOutlineNextSibling ( IPDFOutline* outline)
```

参数：

Outline - 此书签节点将返回下一个（如果存在的话）兄弟节点。

返回值：

如果当前书签节点含有兄弟节点则返回兄弟节点，否则返回空。

4) 页面布局

SetLayoutShowMode

设置页面布局。一个 PDF 文件可以以 n 列 m 行的形式显示。不管 nFacingCount 的数值为多少，如果页面布局模式被设为 MODE_SINGLE,则 ActiveX 窗口将一次显示一页；如果被设为 MODE_CONTINUOUS,则窗口将同时显示相邻的页面。

原型：

```
void SetLayoutShowMode(BrowseMode nShowMode, short nFacingCount);
```

参数：

nShowMode -值可设为如下：

MODE_SINGLE =0.

MODE_CONTINUOUS =1.

nFacingCount -列的数目

返回值：

无:

5) 保存

Save As

保存当前加载的文件到另一个文件

原型：

```
void SaveAs(BSTR FileName)
```

参数：

FileName -指定的要保存的文件名。

返回值：

无

Save

保存当前加载的文件。

原型：

```
void Save ()
```

参数：

无

返回值：

无

SaveToStream

保存当前加载的文件到内存。

原型：

```
IStream * SaveToStream()
```

参数：

无

返回值：

返回一个支持读写 Stream 对象的 IStream 接口, Stream 对象包含了 PDF 文件数据.

6)其它

ExistForwardStack

检测是否存在下一个视图：经常地，当一个用户在浏览 PDF 文件的时候，他可能想退回到上一个浏览的位置。视图被定义为浏览点或者是显示状态。某些的用户动作可能会产生新的视图。比如，一个用户转到一个新页，然后缩放该页，这两个动作将会产生两个新的视图。应用程序可以通过调用这一组方法来让用户方便的在各个不同的视图中跳转。

原型：

```
BOOL ExistForwardStack ();
```

参数：

无

返回值：:

如果存在下一个视图，将会返回 true,否则将会返回 false。

GoForwardStack

转到下一个视图。

原型：

```
void GoForwardStack()
```

参数：

无

返回值：

无

ExistBackwardStack

检测是否存在前一个视图

原型：

```
BOOL ExistBackwardStack ();
```

参数：

无

返回值：

如果前一个视图存在，则将返回 true,否则返回 false.

函数：GoBackwardStack

转到前一个视图。

原型：

```
void GoBackwardStack ();
```

参数：

无

返回值：

无

ShowTitleBar

显示或隐藏标题栏；

原型：

```
void ShowTitleBar(BOOL show)
```

参数：

show -如果参数为 false,则标题栏将不可见。如果为 true,则标题栏将可见。

返回值：

无

ShowToolBar

显示或隐藏工具栏。

原型：

```
void ShowToolBar(BOOL show);
```

参数：

show -如果参数为 false，则工具栏将不可见。如果为 true，则工具栏将可见。

返回值：
空。

ShowToolBarButton

显示或隐藏工具栏上的按钮。

原型：

```
void ShowToolBarButton (short nIndex, short show)
```

参数：

nIndex - 按钮的索引

show - 如果参数值为 0,则按钮将不可见,如果为非 0,则按钮可见。

返回值：
无

函数: ShowBookmark

显示或隐藏书签

原型:

```
void ShowBookmark(short show)
```

参数：

show -如果参数为 0,则书签将不可见,如果为非 0,那么书签将可见。

返回值：
无。

函数: ShowStatusBar

显示或隐藏状态条

原型：

```
void ShowStatusBar(short show);
```

参数：

show - 如果参数为 0,则状态条将不可见,如果为非 0,则状态条将可见。

返回值：
无

GetSelectedText

Get currently selected Text。

原型：

```
BSTR GetSelectedText()
```

参数：

无

返回值：
被选中的文本。

GotoPageView2

转到当前文档的指定位置。

原型:

```
void GotoPageView2(ILink_Dest * link_dest);]
```

参数：

link_dest -ILink_Dest 接口，可以从事件 OnHypeLink 获取到。

返回值：

无。

SetViewRect

该函数可以显示当前 PDF 文档页面中的某个矩形区。这里的坐标系为 PDF 坐标系，而非设备坐标系。该函数将保持 ActiveX 窗口的位置和大小不变，并且调整当前 PDF 页面的位置和缩放比例，使当前 PDF 页面的指定矩形区能够在 ActiveX 窗口中被完全的显示出来。一个典型的应用是：终端用户可以使用鼠标点击并拖曳出一个矩形区然后再释放鼠标，程序将调用 ConvertClientCoodToPageCood 转换客户坐标系到 PDF 坐标系，然后再调用 SetViewRect 以完整的视图来显示指定区域。

原型：

```
void SetViewRect(float left, float top, float width, float height);
```

参数：

left -左上角的水平坐标
top -左上角的垂直坐标
Width -矩行的宽度.
height -矩行的高度

scroll

以单位为像素的 dx,dy 为增量，滚动当前视图

原型:

```
void Scroll(long cx, long cy);
```

参数：

dx -滚动的水平距离。
dy -滚动的垂直距离。

返回值:

无。

OpenFileForPrinter

当你想在不打开文件的情况下打印一个文件，可以调用这个函数.

原型：

```
IPDFPrinter* OpenFileForPrinter(BSTR file_path)
```

参数：

file_path -PDF 文件路径(包括文件扩展名)

返回值:

使用 IPDFPrinter 接口来控制打印机

Hightlight

在当前文档中的指定页上高亮指定的矩形区域.

原型：

```
void Highlight(long nPageIndex, float left, float top, float right, float bottom)
```

参数：

nPageIndex	-页面区域被高亮的页的索引。
left	-在矩形区域的左上角的 X 坐标。
top	-在矩形区域的左上角的 Y 坐标。
right	-在矩形区域的右下角的 X 坐标。
bottom	-在矩形区域的右下角的 Y 坐标。

RemoveAllHighlight

去除当前打开文档中的所有高亮。

原型：

```
void RemoveAllHighlight()
```

参数：

无

返回值：

无

ConvertClientCoodToPageCood

将 ActiveX 控件窗口当中的一个点从客户坐标系转到页面坐标系。

原型：

```
BOOL ConvertClientCoodToPageCood(long nClientX, long nClientY, long* pnPageNum, float* pPageX, float* pPageY);
```

参数：

nClientX	-ActiveX 控件窗口的客户坐标系的 X 坐标
nClientY	-ActiveX 控件窗口的客户坐标系的 Y 坐标。
pnPageNum	-返回给定点落到的 PDF 页面的页号。
pPageX	-返回给定点落到的 PDF 页面的 X 坐标（在 PDF 坐标系）
pPageY	-返回给定点落到的 PDF 页面的 Y 坐标（在 PDF 坐标系）

返回值：

返回值显示了转换是否成功。客户区域包括了 PDF 页面显示区域和灰色的背景区域。如果点落到了灰色的背景区域，则转换失败。

ConvertPageCoordToClientCoord

从 PDF 坐标系转换到 ActiveX 控件窗口的客户区坐标系。

原型：

```
BOOL ConvertPageCoordToClientCoord (long nPageNum, float dPageX, float dPageY, long* pnClientX, long* pnClientY);
```

参数：

nPageNum	-页号
dPageX	- PDF 页面的 X 坐标(PDF 坐标系)
dPageY	-PDF 页面的 Y 坐标 (PDF 坐标系)
pnClientX	-返回 ActiveX 控件窗口客户区的 X 坐标。负值表示点落到了客户区之外。
pnClientY	-返回 ActiveX 控件窗口客户区的 Y 坐标。负值表示点落到了客户区之外。

返回值：

返回值表示了转换是否成功。如果文档没有被正确打开或者如果页号不正确，那么返回值将为 false. 否则，返回值将为 true。

FindFormFieldsTextFirst (*)

查找表单域中的文本。

原型：

Boolean FindFormFieldsTextFirst(BSTR searchstring, boolean bMatchCase);

参数：

Searchstring -想要查找的字符串。

bMatchCase -大小写是否敏感。

返回值：

返回值表示所要查找的字符串是否被找到。

FindFormFieldsTextNext (*)

查找表单域中的下一个匹配的文本。

原型：

boolean FindFormFieldsTextNext()

参数：

无

返回值：:

返回值表示了是否找到了所要查找的字符串。

GetPageText

获取当前加载的 PDF 文件中一页的文本内容。

原型：

BSTR GetPageText(long nPage)

参数：

nPage -要提取文本内容的页号。

返回值：

提取的文本内容。

CountTools

取得在当前版本的 ActiveX 中可用的工具数量。

原型：

short CountTools()

返回值：

工具数目。

GetToolByIndex

取得工具名称。

原型：

BSTR GetToolByIndex(short nIndex)

参数：

nIndex – nIndex 的范围为： 0 <= nIndex < CountTools().

返回值：

工具名称。

RunJavaScript(*)

使用 JavaScript 来控制一些特征。

原型：

BSTR RunJavaScript(BSTR csJS)

返回值：

JavaScript 的返回值。可以是像”0”,”hello”, “The JavaScript runs successfully”...

GetDocPermissions

获取文档的许可标识。

原型：

long GetDocPermissions()

参数：

无

返回值：

表示文档许可标识的 32 位整数。可以参考 PDF Reference 来进一步了解许可信息。如果文档未被保护,将返回 0xffffffff.

ShowDocumentInfoDialog

弹出文档属性对话框。

ShowDocJsDialog (*)

弹出文档的 JavaScript 对话框。

ShowJsConsoleDialog (*)

弹出 JavaScript 控制台对话框。

ExportFormToFDFFile (*)

导出 PDF 表单数据到一个表单数据文件 (FDF)。

原型：

boolean ExportFormToFDFFile(BSTR FDFFileName)

参数：

FDFFileName - fdf 文件路径。

返回值：:

返回值表示操作是否成功。

ImportFormFromFDFFile (*)

从一个表单数据文件导入数据到 PDF 文件。

原型:

boolean ImportFormFromFDFFile(BSTR FDFFileName)

参数：

FDFFileName -fdf 文件路径。 .

返回值：:

返回值表示操作是否成功。

ExportAnnotsToFDFFile (*)

从当前文档导出标注到表单数据文件中去。

原型：

boolean ExportAnnotsToFDFFile(BSTR FDFFileName)

参数：

FDFFileName -fdf 文件路径。

返回值：

返回值表示操作是否成功。

ImportAnnotsFromFDFFile (*)

从表单数据文件导出标注到当前文档来。

原型

boolean ImportAnnotsFromFDFFile(BSTR FDFFileName)

参数;

FDFFileName -PDF 文件路径。

返回值：

返回值表示操作是否成功。

SubmitForm (*)

提交表单数据到指定 URL。

原型:

boolean SubmitForm(BSTR csDestination)

参数：

csDestination -所要提交到 URL。

返回值：

返回值表示操作是否成功。

SetCurrentLanguage

可以动态将 ActiveX 的用户界面语言改成 30 种以上的不同语言。这项特性需要额外的语言文件 (XML 格式) 和 ActiveX 放到一起来实现。如果您想要指定的语种文件, 您可以联系我们: sales@foxitsoftware.com

原型：:

void SetCurrentLanguage(short LanguageID);

参数：

LanguageID -语种标识符。值从 0 到 30,表示不同的语言。

返回值:

无

UnLockActiveX

使用从福昕软件公司接收到的许可码来为 ActiveX 解锁。

原型：

void UnLockActiveX(BSTR lisence_id, BSTR unlock_code)

参数 ::

license_id -从福昕软件公司获取的许可码字符串

unlock_code -从福昕软件公司获取的用来解锁 ActiveX 的

字符串。

返回值 :

空

补充说明 :

如果是试用 ActiveX, 您不需要调用这个函数。但是所有的页面上都会显示试用的标志。对于付费用户, 您应该在调用其它函数之前先调用这个函数。

事件

BeforeDraw

在页面被绘制之前触发。

原型:

BeforeDraw(long dc)

参数 :

dc -设备上下文句柄。

AfterDraw

在页面被绘制之后触发。

原型 :

AfterDraw(long dc)

参数 :

dc -设备上下文句柄。

OnZoomChange

当 Zoomlevel 属性被改变后触发。

原型 : OnZoomChange()

参数 :

无

OnPageChange

当你改变页面时(从一个页跳转到另一个页面)触发。

原型: OnPageChange()

参数 :

无

OnOpenPassword

当你试图打开一个带有密码的 PDF 文件时触发。

原型 : OnOpenPassword (BSTR* password, boolean* cancel)

参数 :

Password -密码字符串
Cancel -如果 Cancel 设为 False,那么它将被一直触发,直到输入了正确的密码。

OnHypeLink

当用户点击一个超链接时触发。

原型: OnHypeLink(BSTR linktype, BSTR linkdata, Link_Dest* dest, boolean* cancel)

参数:

Linktype -包含了超链接类型信息的字符串。

链接类型的字符串可以为:

GoTo 转到当前文档中的其它页, linkdata 是空字符串, dest 包含了控件导航的位置信息。

GoToR 转到存在本地磁盘上的其它 PDF 文件, 如果需要打一个新窗口来浏览新文件, 则 linkdata 中的信息包含了文件名附加上"1", 否则附加上"0". dest 包含了控件导航的位置信息。

Launch 打开外部应用程序, 如果需要打一个新窗口来浏览新文件, 则 linkdata 中的信息包含了文件名附加上"1", 否则附加上"0"

URI 打开一个 URI, linkdata 包含了 URI 字符串。

Cancel 如 cancel 变量被设为 true, 则控件将不会跟随超链接。

linkData 以分隔符':'来分隔, 包含了额外信息的字符串。

OnSearchProgress

当你在搜索文档时触发。

原型: OnSearchProgress(long pageNumber, long pageCount)

参数:

pageNumber -当前被搜索的页号:

pageCount -总共页数。

OnOpenDocument

当打开一个文档时触发。

原型: OnOpenDocument(BSTR filepath)

参数:

filepath -PDF 文件路径。

OnCloseDocument

当你关闭一个文档时触发。

原型: OnCloseDocument(BSTR filepath)

参数:

Filepath -PDF 文件路径。

OnDocumentChange

当 PDF 文档内容改变时触发。

原型: OnDocumentChange()

参数：
无

CustomFileGetSize

当调用 OpenCustomFile 方法打开文档时触发。

原型：CustomFileGetSize(long* size)

参数：

size -[out]接收 PDF 文件长度的指针,将其设为 PDF 文件长度值。

CustomFileGetBlock

当调用 OpenCustomFile 方法打开文档时触发。从指定位置获取数据块，位置从文件开头以字节偏移来计算。位置和大小不会超出文件的长度大小。

原型：CustomFileGetBlock(long pos, long pBuf, long size)

参数：

pos -[in]距文件开头的字节偏移。
pBuf -[out]接收 PDF 数据的数据缓冲区指针。
size -[in]缓冲区大小。

IPDFPrinter 接口

使用 IPDFPrinter 接口，您可以控制打印机和打印输出。

IPDFPrinter 属性

BSTR printerName;

-设置打印机名字用以打印输出。

PrinterRangeMode printerRangeMode;

-设置打印范围，可以被设为：

PRINT_RANGE_ALL = 0,

PRINT_RANGE_CURRENT_VIEW = 1,

PRINT_RANGE_CURRENT_PAGE = 2,

PRINT_RANGE_SELECTED = 3,

short printerRangeFrom;

-您应当先设置 PrinterRangeMode 为 PRINT_RANGE_SELECTED

short printerRangeTo;

-您应当先设置 PrinterRangeMode 为 PRINT_RANGE_SELECTED

short numOfCopies;

-打印的份数。

IPDFPrinter 方法:

PrintWithDialog();

-显示打印设置窗口并打印输出。

PrintQuiet();

-打印输出

SetPaperSize(long paperSize);

-为选定的打印机设置页面大小，有关可用的页面大小请阅读 Windows SDK 文档。

IPDFOutline 接口

IPDFOutline 方法:

void NavigateOutline()

 跳转到书签节点指定的目的地。

BSTR GetOutlineTitle()

 取得书签的标题。

IPDFDocumentInfo 接口

 使用 IPDFDocumentInfo 接口，您可以得到 PDF 文档信息。

IPDFDocumentInfo 属性:

BSTR Author

BSTR Subject

BSTR CreatedDate

BSTR ModifiedDate

BSTR Keywords

BSTR Creator

BSTR Producer

BSTR Title