Foxit ActiveX Tutorial

Contents

Prerequisites

Developer Audience

This document is targeted towards C/C++ developers using the Foxit PDF ActiveX SDK. It assumes the developer is familiar with C/C++ and Microsoft Foundation Classes (MFC).

Supported Environments

Platform	Operating System	Compiler
Windows (32-bit)	Windows 2000/XP or later	Microsoft Visual Studio 6.0 and Visual Studio 2010

Overview

Purpose

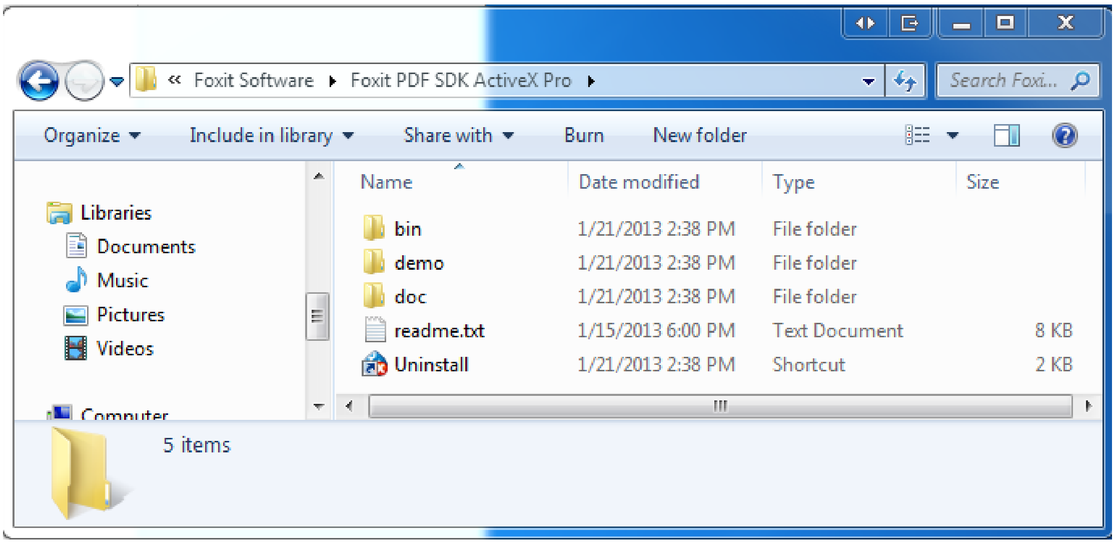
This document covers how to use the Foxit ActiveX SDK. It uses the demo provided by Foxit Corporation as reference for explanation.

Setup

1. Download the free 30 day trial version of ActiveX Professional from http://www.foxitsoftware.com/products/sdk/register.php?product=ActiveX-Professional

2. You will receive an email with a download link for the FoxitPDFSDKActiveX_Pro.msi. Download this msi and double click on it to start the installation.

3. The files are installed under C:\Program Files (x86)\Foxit Software\Foxit PDF SDK ActiveX Pro



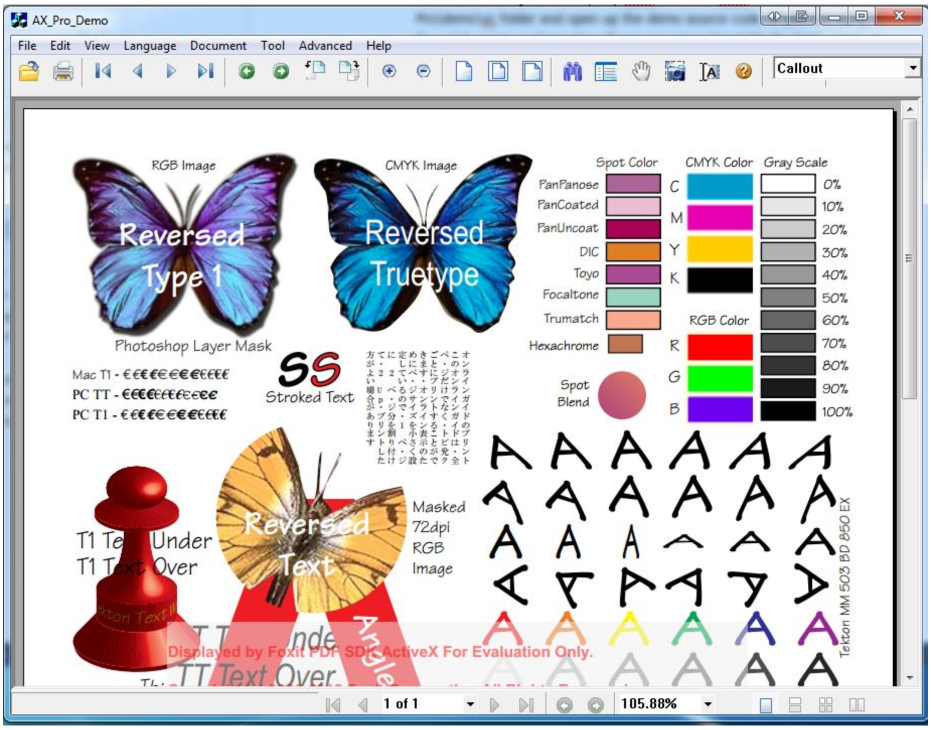
Name	Date modified	Type	Size
bin	1/21/2013 2:38 PM	File folder	
demo	1/21/2013 2:38 PM	File folder	
doc	1/21/2013 2:38 PM	File folder	
readme.txt	1/15/2013 6:00 PM	Text Document	8 KB
Uninstall	1/21/2013 2:38 PM	Shortcut	2 KB

5 items

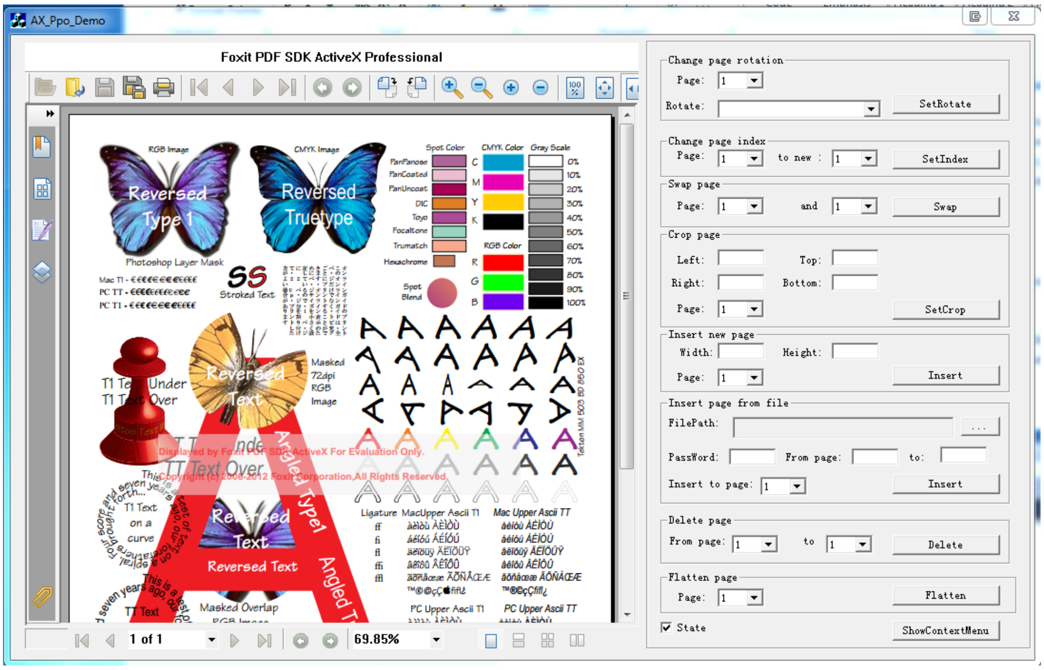
4. Go to the C:\Program Files (x86)\Foxit Software\Foxit PDF SDK ActiveX Pro\demo\vc folder and open up the demo source code. If you are using Visual C++ 6.0, open up demo.dsw. If you are using Visual Studio 2010, open up demo.sln. The instructions and screenshots in this tutorial will reference Visual Studio 2010.

5. The demo.sln file contains 2 demos.

   AX_Pro_Demo – A full featured PDF reader with a drop down list of annotation tools.

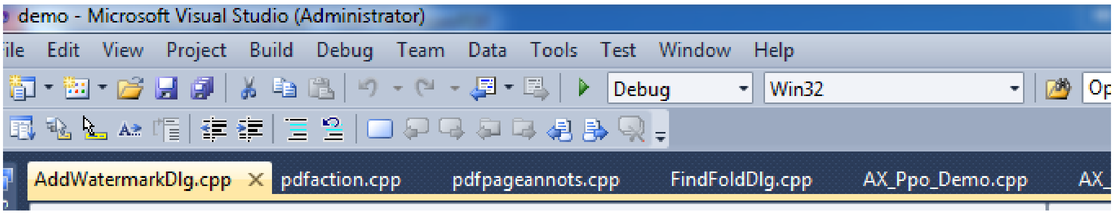


AX_Ppo_Demo – A full featured PDF reader with a panel for page manipulation functions.



6. The remainder of this demo will examine the basic functions developers will start with when using the SDK. Make sure to set AX_Pro_Demo as the startup project. In Visual Studio 2010, go to the Solution Explorer > right click on AX_Pro_Demo > Set as Startup Project.

7. Select to build a Debug version of the demo so you will be able to step through the code.



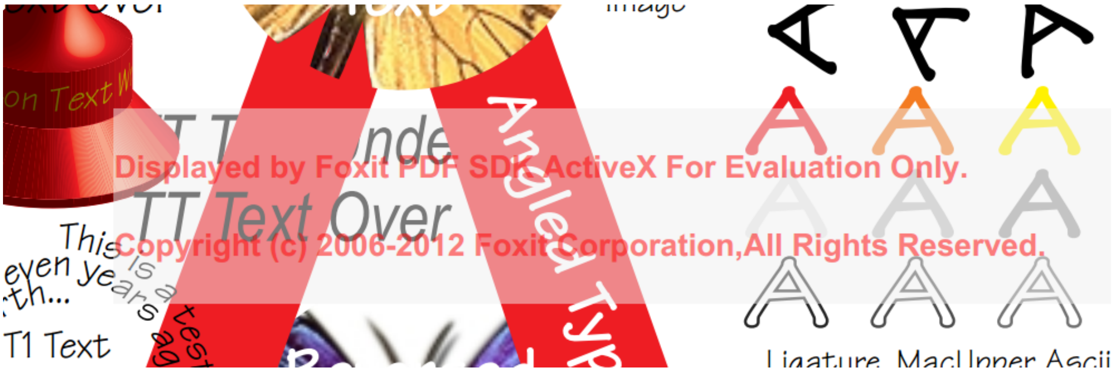
8. Build the demo, Go to Build > Build Solution.
9. Make sure there are no errors in the build output.
10. Run the demo by going to Debug > Start Debugging

Getting Started: Basic PDF Functions

The following sections contain references to the AX_Pro_Demo. To get a full understanding of how the functions works, the user should set breakpoints and step through the code. File names and line numbers for specific function calls are provided.

Unlocking the SDK

The evaluation version of the ActiveX SDK will apply a watermark across any rendered PDF page. Customers who pay for the SDK will receive a license key to remove the watermark.



Displayed by Foxit PDF SDK ActiveX For Evaluation Only.
Copyright (c) 2006-2012 Foxit Corporation,All Rights Reserved.

The license key will contain a license id and an unlock code that you will pass in as parameters to the UnLockActiveX function. There is an example of this in AX_Pro_DemoDlg.cpp,

```
BOOL CAX_Pro_DemoDlg::OnInitDialog()
{
    CDialog::OnInitDialog();

    // code

    m_AX.UnLockActiveX(_T("Licence_ID"),_T("Unlock_code"));

    // code
}
```

Opening a PDF File

Call the OpenFile function to open a PDF file for viewing. There is a working example in AX_Pro_DemoDlg.cpp,

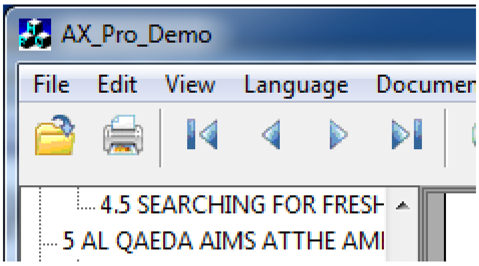
```
void CAX_Pro_DemoDlg::OnFileOpen()
{
    //code

    m_AX.OpenFile(fdg.GetPathName() ,NULL );

    // code
}
```

The user can trigger the OnFileOpen event by going to File > Open > browse to a PDF file > click on Open or by using the file open folder icon.

Go to a specific page

Use the GoToPage function to navigate directly to the page number specified as the parameter. Page indexing is zero based. So call GoToPage(0) to get to the first page.

The AX_Pro_Demo has arrows for navigating to the first, previous, next, and last page of a multipage document. To see these functions in action, run the demo and open a multiple page PDF.



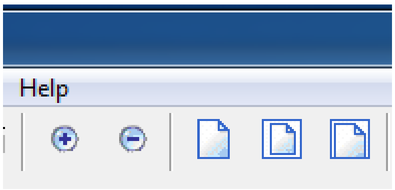
In AX_Pro_DemoDlg.cpp, you will see the corresponding events that are triggered when the arrow buttons for navigating pages are pressed. You will see the GoToPage function is called with the appropriate index value.

```
void CAX_Pro_DemoDlg::OnFirstPage()
{
    m_AX.GoToPage(0);
}
```

```
void CAX_Pro_DemoDlg::OnLastPage()
{
    int count = m_AX.GetPageCount();
    m_AX.GoToPage(count-1);
}


void CAX_Pro_DemoDlg::OnNextPage()
{   int count = m_AX.GetPageCount();
    int current = m_AX.GetCurPage();

    if(current+1 <= count-1)
        m_AX.GoToPage(current+1);
}

void CAX_Pro_DemoDlg::OnPrevPage()
{
    int count = m_AX.GetPageCount();
    int current = m_AX.GetCurPage();

    if(current-1 >= 0)
        m_AX.GoToPage(current-1);
}
```

Zoom in and out of a page

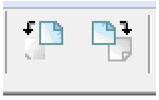
Call SetZoomLevel to zoom in and out of the PDF page. The AX_Pro_Demo has menu buttons for zoom in, zoom out, actual size, fit page, and fit width. All of these buttons trigger events that call SetZoomLevel.



```
void CAX_Pro_DemoDlg::OnZoomIn()
{
    zoomfactor = m_AX.GetZoomLevel();

    if(zoomfactor + 10 > 6400)
        zoomfactor = 6400;
    else
        zoomfactor = zoomfactor + 10;

    m_AX.SetZoomLevel( zoomfactor );
}

void CAX_Pro_DemoDlg::OnZoomOut()
{
    zoomfactor = m_AX.GetZoomLevel();

    if(zoomfactor - 10 > 0)
        zoomfactor = zoomfactor - 10;

    m_AX.SetZoomLevel( zoomfactor );
}

void CAX_Pro_DemoDlg::OnActualSize()
{
    m_AX.SetZoomLevel(0);
}

void CAX_Pro_DemoDlg::OnFitPage()
{
    m_AX.SetZoomLevel(1);
}

void CAX_Pro_DemoDlg::OnFitWidth()
{
    m_AX.SetZoomLevel(2);
}
```

Rotate a Page

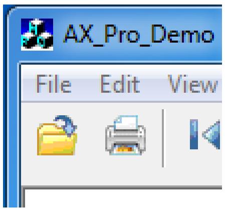
Call SetRotate to rotate the PDF page. The AX_Pro_Demo has menu buttons for rotate right and rotate left. These buttons trigger events that call SetRotate.



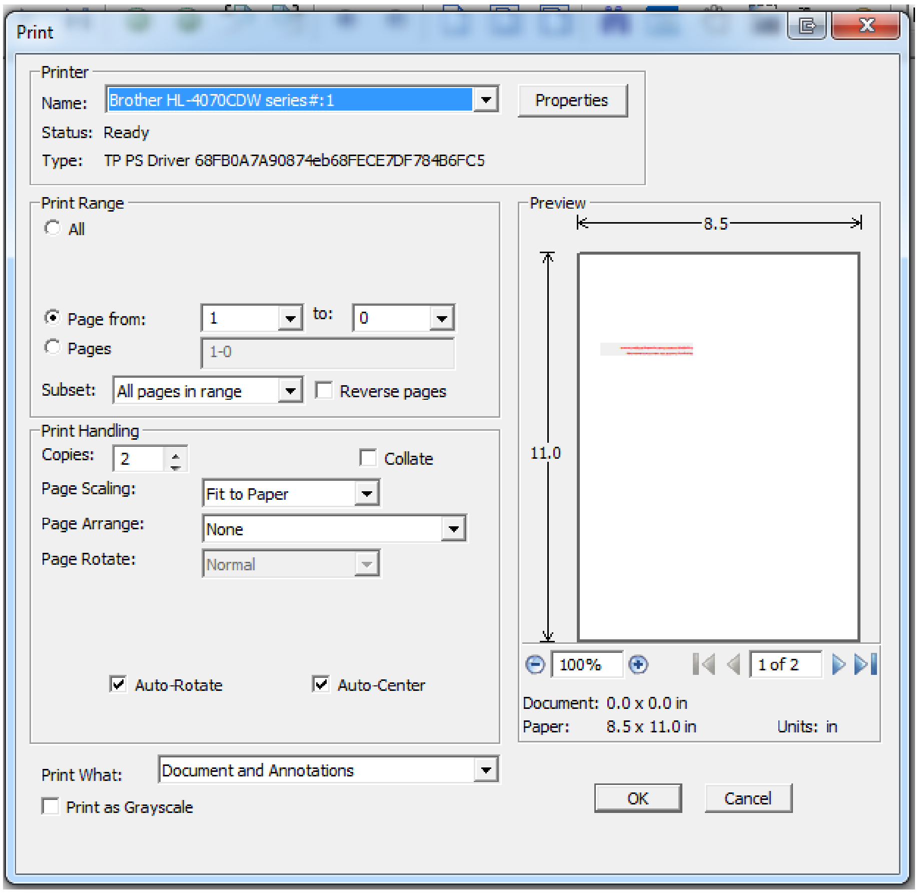
```
void CAX_Pro_DemoDlg::OnRotateRight()
{
    if(rotatefactor == 3)
         rotatefactor = 0;
     else
         rotatefactor++;
     m_AX.SetRotate(rotatefactor);
}

void CAX_Pro_DemoDlg::OnRotateLeft()
{
     if(rotatefactor == 0) rotatefactor=3;
    else rotatefactor--;
     m_AX.SetRotate( rotatefactor );
}
```

Print a PDF document

Call PrintWithDialog to bring up the print dialog page. On that page you can manually adjust printer settings. Using the ActiveX SDK you can also programmatically set printer settings. Clicking on the printer icon



or going to File > Print will trigger the OnFilePrint() event that calls the PrintWithDialog function to bring up the print dialog settings,



Print

Printer

Name: Brother HL-4070CDW series#:1

Status: Ready

Type: TP PS Driver 68FB0A7A90874eb68FECE7DF784B6FC5

Print Range

○ All

● Page from: 1 to: 0

○ Pages 1-0

Subset: All pages in range □ Reverse pages

Print Handling

Copies: 2 □ Collate

Page Scaling: Fit to Paper

Page Arrange: None

Page Rotate: Normal

☑ Auto-Rotate ☑ Auto-Center

Print What: Document and Annotations

□ Print as Grayscale

Preview

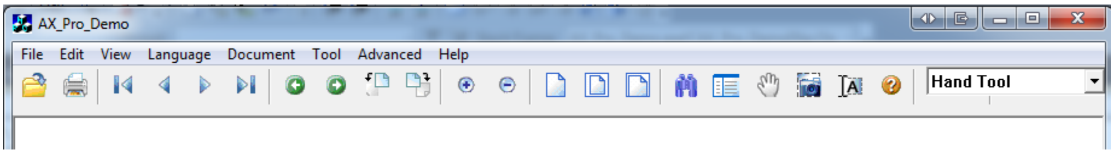
Document: 0.0 x 0.0 in

Paper: 8.5 x 11.0 in Units: in

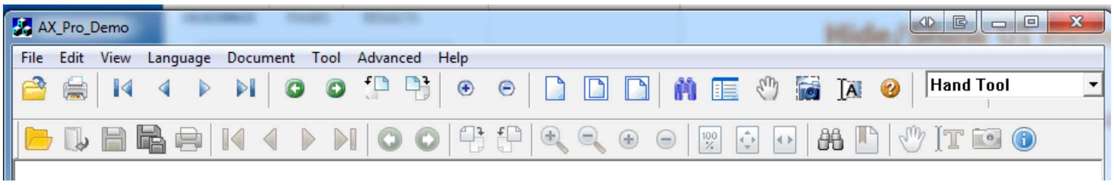
```
void CAX_Pro_DemoDlg::OnFilePrint()
{
    CPDFPrinter printer = m_AX.GetPrinter();
    if (!printer)
    {
        return;
    }
    //printer.SetPrinterName( "foxit PDF" );
    printer.SetPrinterRangeMode( 3 );
    printer.SetPrinterRangeFrom( 2 );
    printer.SetPrinterRangeTo( 3 );
    printer.SetPaperSize( 8 );
    printer.SetNumOfCopies( 2 );
    printer.PrintWithDialog();
}
```

Hide/Show UI elements

The ActiveX SDK provides functions for modifying the user interface. For example, the ShowToolBar() function takes a boolean parameter. This is what the user interface looks like when ShowToolBar(FALSE) is called by the application code,



ShowToolBar(TRUE) adds the toolbar to the user interface

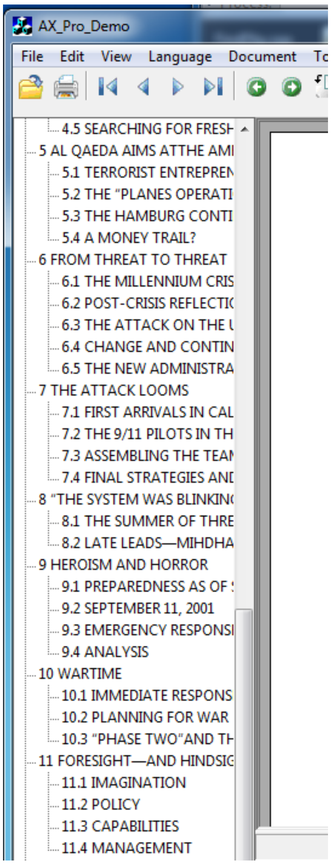


In AX_Pro_DemoDlg.cpp, the ShowToolBar function is called with FALSE as the default setting. As an exercise, the user can change m_ToolState to be TRUE to enable the toolbar when the demo is run.

```
BOOL CAX_Pro_DemoDlg::OnInitDialog()
{
    // code

    m_ToolState = FALSE;

    // code

    m_AX.ShowToolBar( m_ToolState );

    // code
}
```

Iterate the tree outline of the PDF document

If the user wants to build a table of contents for a PDF document, they can use the GetOutlineFirstChild() and GetOutlineNextSibling() functions to iterate through the outline tree of the PDF. Here’s an example of a TOC using the AX_Pro_Demo,



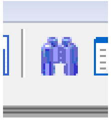
4.5 SEARCHING FOR FRESH
5 AL QAEDA AIMS ATTHE AMI
5.1 TERRORIST ENTREPREN
5.2 THE "PLANES OPERATI
5.3 THE HAMBURG CONTI
5.4 A MONEY TRAIL?
6 FROM THREAT TO THREAT
6.1 THE MILLENNIUM CRIS
6.2 POST-CRISIS REFLECTIC
6.3 THE ATTACK ON THE U
6.4 CHANGE AND CONTIN
6.5 THE NEW ADMINISTRA
7 THE ATTACK LOOMS
7.1 FIRST ARRIVALS IN CAL
7.2 THE 9/11 PILOTS IN TH
7.3 ASSEMBLING THE TEAM
7.4 FINAL STRATEGIES AND
8 "THE SYSTEM WAS BLINKIN
8.1 THE SUMMER OF THRE
8.2 LATE LEADS—MIHDHA
9 HEROISM AND HORROR
9.1 PREPAREDNESS AS OF S
9.2 SEPTEMBER 11, 2001
9.3 EMERGENCY RESPONSI
9.4 ANALYSIS
10 WARTIME
10.1 IMMEDIATE RESPONS
10.2 PLANNING FOR WAR
10.3 "PHASE TWO"AND TH
11 FORESIGHT—AND HINDSIG
11.1 IMAGINATION
11.2 POLICY
11.3 CAPABILITIES
11.4 MANAGEMENT

In AX_Pro_DemoDlg.cpp in the CAX_Pro_DemoDlg::OnFileOpen() function there is code that shows how the TOC is generated when the file is first open in the demo. To get a closer understanding, it is recommended that the developer set a breakpoint at the beginning of this function and step through. See where GetOutlineFirstChild and GetOutlineNextSibling are called.

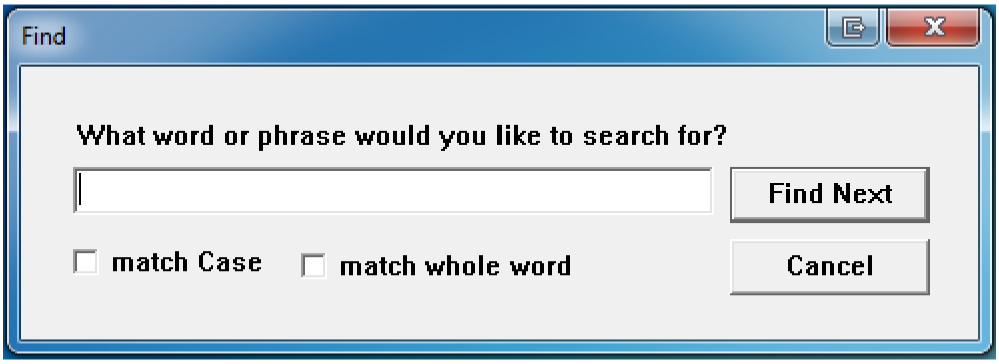
```
void CAX_Pro_DemoDlg::OnFileOpen()
{
    // code

    CPDFOutline outline;
    outline = m_AX.GetOutlineFirstChild( outline.m_lpDispatch );

    if (outline != NULL)
    {
        outline = NULL;
        ProcOutline(outline,NULL);
    }

    OnMENUITEMexpandtree();
    if (m_outlinetree.GetCount()>0)
    {
        ShowBookMark();
        m_bhidebookmark = TRUE;
    }
    else
    {
        HideBookMark();
        m_bhidebookmark = FALSE;
    }
}
```

Search

The AX_Pro_Demo has a search icon in the toolbar that allows a user to search through a PDF for specific words or phrases.



This Find dialog box appears when the icon is clicked on,



What word or phrase would you like to search for?

The FindFirstEx() function finds the first occurrence of the string in the PDF document. The parameters are the search string, a boolean to indicate if the user wants to match case, a Boolean to indicate if the user wants to match the entire word. In FindDlg.cpp in the CFindDlg::OnOK() function there is an working example of FindFirstEx() and FindNext().

```
void CFindDlg::OnOK()
{
    UpdateData(true);
    If (TRUE == m_bFirst)
    {
        // code to initialize the var parameter

        m_pAX->FindFirstEx(var, m_check1, m_check2);

        // code
    }
    else
        //FindNext(BOOL) TRUE:down search, FALSE:up search
        m_pAX->FindNext(TRUE);
}
```