# Acrobat Developer FAQ

*Adobe Developer Technologies*

**Revised: November 12, 1999**

## Version History

| | | |
|---|---|---|
| 05 May 1998 | Acrobat Developer Support | First version. |
| 09 June 1998 | Acrobat Developer Support | Second version. |
| 16 July 1998 | Acrobat Developer Support | Third version. |
| 9 September 1998 | Acrobat Developer Support | Minor edit. |
| 29 October 1998 | Acrobat Developer Support | Minor edit. |
| 15 December 1998 | Acrobat Developer Support | 4.0 version. |
| 7 January 1999 | Acrobat Developer Support | 4.0 version. |

## How to Use This FAQ

You can choose to browse this FAQ by the topic of your choice (Acrobat Developer Topics), question and answer format (Acrobat Developer Q&A), or by viewing the PDF version. The links from the Acrobat Developer Topics and Acrobat Developer Q&A go to the corresponding discussion items. Links to sample code and documentation are set up to go to their corresponding Web links. In other words, when you click on a documentation or sample code link within this document, it takes you to a Web link that takes you to the appropriate URL. The Web links are grouped in three categories:

- Links to Documentation
- Acrobat Software Developer Resources
- Samples (Pre-compiled code)

You can use these three pages independently of the discussion items. The documentation links and sample code links do not download their corresponding files. Those links will just take you to the appropriate URL where you can then find the particular document or sample by name.

Browse by Acrobat Developer Topics

Browse by Acrobat Developer Q&A

# Acrobat® Developer Topics

# Acrobat Developer Q & A

## Basic Acrobat SDK Questions

### What Is Acrobat?

Acrobat 4.0 software allows you to convert electronic files created in your favorite applications into Portable Document Format (PDF) and distribute them via the Web, intranet, CD-ROM, or e-mail. The product suite includes the following applications:

- Acrobat Reader for viewing, navigating, and printing PDF documents.
- Acrobat for adding navigational links, annotations, and security options, in addition to the functionality provided by Acrobat Reader.
- Acrobat PDFWriter for creating PDF files from your favorite business applications using only the Print command.
- Acrobat Distiller® for creating PDF files from your favorite PostScript® applications (including desktop publishing software).
- Acrobat Catalog for creating full-text indices, searchable with Acrobat Search.
- Acrobat Capture® plug-in with OCR (Optical Character Recognition) for creating text-searchable PDF documents from scanned paper originals.

For further information about the Acrobat product, see the Acrobat Main Page.

### What Is the Acrobat SDK?

The Acrobat Software Development Kit (SDK) is a “toolbox” that contains header files, type libraries, simple utilities, sample code, and documentation. These tools help you design and develop Acrobat plug-ins as products or as part of integration with your application. The plug-ins control the Acrobat product suite, using Interapplication Communication (IAC) interfaces such as Apple events, OLE, OLE automation, and DDE.

The Acrobat SDK is available on the Adobe public Web site. The Acrobat SDK is also included on a CD with membership to the Adobe Solutions Network (ASN) Developer Program. We encourage you to join the ASN Developer Program to receive developer technical support, marketing assistance, and other benefits.

To access the Acrobat SDK, see the Acrobat SDK Main Page. For more information on the ASN Developer Program and how to join, see the ASN Developer Program Home Page.

### What Does the Acrobat SDK Consist Of?

The SDK consists of documentation (technical notes), samples (example source code), and header files (to compile the samples).

There are also two object code libraries available:

1) The Forms Toolkit is available for free at http://partners.adobe.com/asn/developer/acrosdk/forms.html. See the Using Acrobat Forms and the FDF Toolkit section for more information.

2) The Adobe PDF Library is no longer available for open licensing. For more information, see Licensing and Distribution Issues for Acrobat.

Related Documentation:

“Adobe Acrobat Software Development Kit Contents Description”

## What Parts of the Acrobat SDK Are Not Freely Available on the Web?

The Adobe PDF Library is not free and is not available on the Adobe public Web site. For more information, see Licensing and Distribution Issues for Acrobat.

Technical support for the Acrobat SDK ("Acrobat Developer Support") is also not included for free. An Acrobat Program Membership is required to obtain support for the Acrobat SDK.

## What Do I Need to Download from the Web to Get the SDK?

The SDK is made up of many different technical documents and samples for many different operating systems. You will have to determine what it is you want to do and then determine if there is a sample that gets you pointed in the right direction. You can download the installer from the Web site to obtain the entire Acrobat SDK, or you can download each technical note separately. The sample files and documentation are what comprise the Acrobat SDK.

See the Acrobat SDK Main Page to access the publicly-available Acrobat SDK. Alternatively, you can join the Adobe Solutions Network (ASN) Developer Program and install the Acrobat SDK directly from the Acrobat SDK CD.

For information on what is not included on the public Acrobat SDK Web site, see What Parts of the Acrobat SDK Are Not Freely Available on the Web?.

For information on where to begin with the Acrobat SDK, see Where Do I Start with the Acrobat SDK?.

## What Is a Plug-in?

Plug-ins are dynamically-linked extensions to the Acrobat viewers. They can hook into the user interface in a number of ways and can register to be called when a variety of events occur in the application.

An Acrobat plug-in is a program written in ANSI C/C++ that uses the Acrobat API and adds functionality to Acrobat or Acrobat Reader. A plug-in program file goes into a plug-in folder/directory (usually named "Plug-ins") and is initialized upon startup of the Acrobat viewers.

Plug-ins are:

- DLLs under Windows
- Code fragments on Macintosh PowerPC
- Shared libraries under UNIX

Note: Under Windows, plug-in names must end in `.API`, not `.DLL`. Under UNIX, plug-in names must end in `.API` and the plug-in path must be specified correctly in the `.acrorc` file. In UNIX, the first time you open Acrobat or Reader, a preferences file named `.acrorc` is created in your `$HOME` directory. For more information on the `.acrorc` file, see the Acrobat 4.0 User Guide included with the Acrobat product suite.

## What Are Some Common Implementations Using the Acrobat SDK?

Providing a product out of the box that is suitable for everyone's needs is not possible. The Acrobat SDK is aimed at those who want to customize or modify how Acrobat looks and operates. The Acrobat SDK offers the information necessary to customize and extend Acrobat's functionality to meet your own needs. The SDK documents and demonstrates what the Acrobat API is capable of and how to use it.

Some common uses of the Acrobat SDK include:

- Viewing a PDF file in your own application's window using Acrobat. (See Integration of Acrobat Viewers for PDF Display in an External Window for more information.)
- Using Acrobat to implement a context-sensitive help system.
- Writing a plug-in to add or change menu bars, menu items, or tool-buttons (that is, customizing the user interface). See the "Template" sample plug-in for an example of customizing the user interface.
- Controlling an Acrobat or Reader session. See the "DDEClnt" sample plug-in for an example of communicating with an Acrobat plug-in from an external application using DDE.
- Adding annotation types. See the "Stamper" sample plug-in for an example of adding an annotation type.
- Adding action types (such as going to a URL in a Web browser). See the "Balloon" sample plug-in for an example of adding an action type.

## Where Do I Start with the Acrobat SDK?

The first place to start is with the "Getting Started Using the Adobe Acrobat Software Development Kit (SDK)" document. The Adobe Acrobat products have an extensive API for integration. This SDK includes a large number of sample applications and plug-ins to demonstrate how to use these APIs. Since finding the right examples for your application can be difficult, the "Samples Roadmap" helps direct you to the samples most appropriate for your application.

The "Samples Roadmap" is intended to be a starting point for your integration with Acrobat. There are many other ways for integrating your application and many other types of applications than those mentioned in the roadmap document.

At this point, it is a good idea to read the README file for the Acrobat product, which is included on the Acrobat CD, and the "Release Notes for Adobe Acrobat 4.05 SDK".

Once you find out if what you're trying to do is possible, the first place to look is at the "overview" documents that break down the Acrobat API into two categories:

- Plug-in API. The technical note "Acrobat Core API Overview" discusses the programming interface for developing a plug-in for Acrobat and Acrobat Reader.
- Interapplication Communication (IAC) API. The technical note "Acrobat Interapplication Communication Overview" discusses implementing OLE Automation with Acrobat, DDE (with Acrobat and Reader), and AppleScript and Apple events for Mac OS.

You should also look at the "Samples" document, which gives you an overview of what each sample does.

Related Documentation:

"Release Notes for Adobe Acrobat 4.05 SDK"

## How Is the API Organized?

The API organizes both the Acrobat viewers and PDF files into objects that plug-ins can manipulate. These objects have types, encapsulate their data, and are controlled by methods. This object-orientation is a conceptual model, implemented using a standard C API.

The objects are organized into a hierarchy containing three layers:

- The Acrobat Viewer layer (AcroView or AVModel) deals with the Acrobat viewer. Its methods allow plug-ins to manipulate components of the Acrobat viewer application itself, such as menus and menu items.
- The Cos Object System (Cos) layer provides access to the building blocks used to construct documents. Its methods allow applications to manipulate the low-level data in a PDF file, such as the dictionary and string objects.
- The Portable Document layer (PDModel) provides access to components of PDF documents. Its methods allow applications to manipulate document components such as fonts and document pages.

  PDFEdit treats page content as a list of objects whose values and attributes can be modified. PDFEdit allows you to read, write, edit, and create page contents and page resources, which may contain fonts, images, extended graphics states, and so on.

  PDSEdit. PDF files are best known for representing a document's physical layout. Objects are drawn in the order that they appear in the display list, painting from back to front. In the past, information about the structure of the document's content was lost when the PDF file was generated from the original application. Acrobat 4.0 and PDF1.3 contain features that store structure in PDF files.

In addition to this hierarchy, there are two other groups of functions available to plug-ins:

- The Acrobat Support layer (AS layer) provides platform-independent utility functions, such as file handling mechanisms. The latter is important to document management systems that allow users to directly open files, and to online systems that provide dial-up access to remote documents.
- Platform-specific utilities, available only on specific platforms. For example, the Macintosh API provides methods for converting rectangles between Acrobat device coordinates and Macintosh screen coordinates, and for hooking into the Acrobat viewer's Apple event handling loop. The UNIX API provides methods to read resources and obtain information about the environment in which the plug-in is running.

See the "Acrobat Core API Overview" for more information on how the Acrobat API is organized.

## What Is Included in the ASN Developer Program and How Do I Join?

The Adobe Solutions Network Developer Program includes marketing tools and technical tools. For complete information on benefits the ASN Developer Program, visit:

http://partners.adobe.com/asn/developer/main.html

In the United States, Canada, and outside of Europe and Japan, contact:

ASN Developer Program
Adobe Systems Incorporated
801 North 34th Street
Seattle, WA 98103
Program information: 800.685.3510 or 206.675.6145
Fax:206.675.6872
Membership questions, e-mail: asndeveloper@adobe.com

In Europe, contact:

Adobe Solutions Network
PO Box 12356
Edinburgh EH11 4GJ
Scotland, United Kingdom
Phone: +44.131.458.6800
Fax: +44.131.458.6801
E-mail: euroasn@adobe.com

In Japan, contact:

Developer Support
Adobe Systems Co., Ltd.
Yebisu Garden Place Tower
4-20-3 Ebisu, Shibuya-ku
Tokyo 150-6017 Japan
Phone: +81.3.5423.8169
Fax: +81.3.5423.8204
http://www.adobe.co.jp

Technical Support:

Support for the Adobe Acrobat SDK is available to Adobe Solutions Network (ASN) Developer Program Premium members or may be purchased (except in Japan).

Technologies Supported:

Interapplication Communication (IAC), including Apple events, OLE, and DDE; Adobe-generated Portable Document Format (PDF); pdfmark operator; command-line and initialization file interfaces, and the Acrobat API.

Regular Updates and Upgrades:

Members receive updates to technical notes, development tools, and development software, as well as upgrades to the Acrobat SDK.

Acrobat Software Development Kit (SDK):

See What Is the Acrobat SDK? and What Does the Acrobat SDK Consist Of?

## Licensing and Distribution Issues for Acrobat

### Acrobat Licensing

Some of the Acrobat products may be licensed for commercial redistribution:

1. Acrobat (fee-based, license required)

   Acrobat is licensed as part of the Acrobat 4.0 product.

   For OEM's and Integrators, Adobe offers special pricing for both the Acrobat 4.0 application and the Acrobat viewer only. For more information, send e-mail to integrator-sales@adobe.com.

2. Acrobat Reader (freely distributable)

   You may freely distribute Acrobat Reader, subject to certain restrictions, as set forth in the license agreement. However, you must use the installer that comes with Reader. You may call the Reader installer (InstallShield 3) from your installer, but the installer must remain intact and the End User License Agreement (EULA) must be displayed during installation. You may customize the Reader installer to the extent documented in the `abcpydoc.ini` file. This file is located on the Acrobat Reader 4.0 CD. See Distributing Acrobat Reader for details.

You may also freely distribute Acrobat Reader via CD or other means, such as e-mail or the web. Please see Creating a CD to Distribute Acrobat Reader for details.

Also, if you have an interest in developing a plug-in for Acrobat Reader, see Development of Plug-ins for Acrobat Reader.

3. Acrobat (fee-based, license required)

   Acrobat includes:

   - Acrobat
   - Acrobat PDFWriter
   - Acrobat Distiller
   - Acrobat Catalog
   - Acrobat Capture plug-in, and other plug-ins

4. Acrobat Capture (fee-based, license required)

   Acrobat Capture 2.0 software is a stand-alone product and is not included as part of the Acrobat 4.0 product. It is licensed as a separate product. Low-volume Capture requires a hardware dongle that counts the pages processed; dongles are available with various page limits. However, there is a Capture Plug-in included with the Acrobat product that allows low volume processing of image files into PDF.

   For OEM's and Integrators, Adobe offers special pricing for Acrobat Capture. For more information, send e-mail to integrator-sales@adobe.com.

5. Acrobat Toolkits/Libraries

   a) FDF Toolkit (free):

   The FDF Toolkit is free and available on the Adobe public Web site at http://partners.adobe.com/asn/developer/acrosdk/forms.html.

   See Using Acrobat Forms and the FDF Toolkit for more information.

   b) Adobe PDF Library (fee-based, license required):

   The Adobe PDF Library is an object code library that can be linked to your application. One of the many things it can do is generate PDF.

   The Adobe PDF Library is not free and is not available on the Adobe public Web site. With the introduction of Acrobat 4.0, Adobe has replaced the Adobe PDF Library licensing program. The new licensing program will focus on products that are strategic to Adobe's marketing plans. The new program will also carry significantly redefined licensing fees, terms, and conditions.

   If you are interested in licensing the PDF Library, please complete the PDF Library Licensing Request form at:

   http://partners.adobe.com/asn/developer/acrosdk/pdfLibrary.html

   If your request meets Adobe's criteria for PDF Library licensing, you will be contacted directly by an Adobe representative to discuss the terms and conditions under which we license the PDF Library. At a minimum, it will take 30 days to review your request.

   We are no longer licensing PDF Library 1.0.

For more information on the Adobe PDF Library, see PDF Generation.

c) Placed PDF Library (no longer licensed):

The Placed PDF Library is an object code library that allows an application to place a single page PDF inside a document. It requires special licensing.

We are no longer licensing Placed PDF Library.

The functionality of the Placed PDF Library has been incorporated into the PDF Library. There is no longer a separate library specifically for handling PDF files in a manner similar to EPS files. Although code changes will be minimal, developers updating applications previously written for Placed PDF will need to update their applications to use the new methods in the PDF Library.

For more information on the distribution of Acrobat, see the Acrobat Main Page. If you have Acrobat Technology licensing questions, please send e-mail with your contact information and some information about your product to integrator-sales@adobe.com. The appropriate person will contact you directly.

## What Is Supported And Unsupported in the Acrobat SDK?

In general, Acrobat Developer Support supports software development with the Acrobat Application Programming Interface (API), as documented in the Acrobat Software Development Kit. (Only registered members of the Adobe Acrobat program are supported.) Acrobat Developer Support does not support use of the product that does not involve the Acrobat API. Any non-programmatic issues should be directed to Adobe Acrobat Technical Support. Due to limited resources, a specific range of development activities using the Acrobat SDK are supported as well. The supported activities include those for which the product is designed, tested, and licensed. See Other Adobe Developer Resources for information on getting help with unsupported issues.

### Supported Development Environments

The Acrobat API and SDK is designed and tested to work with specific development environments. Development in other environments is unsupported. In general, Adobe supports the same operating systems for development as is supported for use of the application. The supported operating system for each application can be found in the product information on the Adobe public Web site.

The supported development environments are:

- 32-bit Windows® == Microsoft® Visual C++ for plug-in development and Visual Basic for OLE automation
- Mac OS == Metrowerks CodeWarrior
- UNIX == gcc

For more information, including the supported versions of the compilers mentioned above, check the “Release Notes for Adobe Acrobat 4.05 SDK”.

### Supported Use of the Software Development Kit

Support of the Acrobat SDK includes development with the public APIs to the Acrobat products in which it is designed, tested, and licensed to be used. Check the overview documentation (listed immediately below) for the SDK, as well as the following documents for a discussion of intended uses of the Acrobat SDK.

Relevant Documentation:

"Getting Started Using the Adobe Acrobat Software Development Kit (SDK)"
"Acrobat Core API Overview"
"Acrobat Interapplication Communication Overview"
"Acrobat API Development"
"Samples Roadmap"

## Unsupported Use of the Software Development Kit

The following is a list of unsupported uses of the Acrobat Software Development Kit. Most activities are given a descriptor of "Technically Infeasible" and/or "Contrary to Licensing."

1. Use of any Acrobat product in a multithreaded way.

   Technically Infeasible

   At this time, no Acrobat applications are multithreaded or thread-safe. Any multithreaded access of the Acrobat API is likely to crash or hang the application.

   For more information pertaining to multithreaded use of Acrobat software, see Multithreading and Acrobat.

2. Use of any Acrobat product as a server process accessed by multiple clients, unless it is specifically stated in product documentation as designed and licensed for such purpose.

   Technically Infeasible | Contrary to Licensing

   Unless specifically stated in the product documentation and licensing agreement, Acrobat products are not designed, tested, or licensed for this purpose. This use is in violation of the licensing restrictions, as described in the End User License Agreement displayed during product install and not supported by Acrobat Developer Support.

3. Use of Distiller as an NT service.

   Technically Infeasible

   Acrobat Distiller requires the ability to open a window on the desktop to run. It is not possible to use Distiller as an NT service. This feature request has been noted.

4. Use of MenuItemExecute to bring up Acrobat dialog boxes when a PDF file is displayed in an external window using OLE Automation.

   Technically Infeasible

   Due to problems of managing window focus, using the Acrobat dialog boxes (using MenuItemExecute) when a PDF file is displayed in an external window using OLE Automation is not supported. The actions executed by the dialog box may or may not affect the intended PDF file and there can be problems of windows not redrawing properly. The only supported workaround is to use the OLE Automation methods or to develop a plug-in to achieve any functionality not available in the OLE API to Acrobat.

5. Use of the "Portable Document Format Reference Manual" to develop a third-party application that writes PDF files without the use of Adobe Acrobat products.

   Due to limited resources, we do not have the ability to assist developers in creating their own PDF generation capabilities without the use of Adobe products. The "Portable Document Format Reference Manual" is the best resource for this kind of development. The use of Adobe products to create PDF files as a benchmark for your own development is also a good resource. Acrobat Developer Support will not debug PDF files created with non-Adobe products. Questions regarding the completeness or accuracy of the “Portable Document Format Reference Manual” will be answered.

6. Interapplication Communication (IAC) between a plug-in and a third-party application.

   Interapplication Communication between a plug-in and a third-party application does not differ significantly from interapplication communication between two stand-alone applications. Documentation for your development platform’s API and development environment are the best resources for this type of development. The Acrobat SDK contains two samples that can serve as examples. "DDEClnt" demonstrates DDE communication between a stand-alone application and a plug-in. "ExtrnWin" uses Windows messaging to communicate with a stand-alone application.

   One typical difficulty for developers occurs when a multithreaded stand-alone application communicates with a plug-in. See Multithreading and Acrobat for more information.

7. Use of platform API methods or API methods of applications other than Acrobat.

   Acrobat Developer Support supports the API to the Acrobat product suite. Questions regarding the use of platform API methods should be directed to the manufacturer of your operating system.

8. Use of the ActiveX Control or Netscape plug-in to display a PDF file in an external application besides Internet Explorer or Netscape.

   The methods used by Netscape and Internet Explorer have been built into the product and are only tested for use with these applications. To use the Netscape plug-in in your application, you would need to provide the full Netscape plug-in architecture (Netscape documents their plug-in architecture). Using the Internet Explorer ActiveX Control is easier, but requires you to have Internet Explorer 3.01 or later installed on your system. The problem is that these interfaces (Netscape plug-in and Internet Explorer ActiveX Control) will change from version to version to take advantage of the browser capabilities and changing APIs. We do not suggest development with these interfaces and offer no support of it. Adobe has no documentation available regarding ActiveX (OCX) development with Acrobat.

9. Automating importing image files using the Import and Capture plug-ins to Acrobat.

   Technically Infeasible

   The Import and Capture plug-ins to Acrobat do not provide an API that allows them to be called from a plug-in or another application. Executing the Import Image menu item with MenuItemExecute brings up a dialog box requiring user input. It is unsupported and technically infeasible to attempt to automate this process. The supported method of programmatically converting image files to PDF files is with the use of Acrobat Capture 2.0, which is an OLE Automation server. For smaller applications, there are third-party applications available that can convert simple TIFF files to PDF. For information about other developer resources, see Other Adobe Developer Resources.

## Other Adobe Developer Resources

For information on using or installing Adobe Acrobat products, start at the Adobe Customer Support Page, which contains a searchable database of Technical Support issues, platform-specific technical guides, user forums, and Acrobat Technical Support phone numbers. See also http://www.planetpdf.com for PDF-related resources, hints, tips, and tricks.

If you do not find your answer online, you can contact Adobe Acrobat Technical Support:

| | |
|---|---|
| Windows | 206-675-6304 |
| Mac OS | 206-675-6204 |
| UNIX | 206-675-6404 |

For International customers, see http://www.adobe.com/supportservice/intlsupport.html.

### Places of Interest to Acrobat Developers on the Adobe Web Site

Adobe Acrobat Resources — Looking for plug-ins and applications available from Adobe or third parties, as well as links to pages on how to use the available plug-ins, check out the Acrobat Resources Page.

Acrobat Main Page — White papers and case studies of what can be done and what has been done with Adobe Acrobat.

Information on Partner Resources — Acrobat/PDF service providers.

PDF NewsGroups — News Groups and Electronic Mailing Lists.

### Other Developer Resources Provided by Outside Companies

There are a number of resources made available by PDFZone that may be of help to those developing with and using the Acrobat products. These resources include forums, mailing lists, and an archive of previous mail.

Information on Partner Resources — Acrobat/PDF service providers.

BCL-Computers — Provide tools that allow transfer of PDF objects (such as, tables) into Microsoft Word/Excel, and so forth.

## Acrobat Versus Acrobat Reader: What Are The Differences?

Acrobat is the "full-featured" development viewer. The entire Acrobat API is available to Acrobat.

The Acrobat Reader API is limited technically and legally. Technical limitations are documented in the "Acrobat Core API Overview" and the "Acrobat Interapplication Communication Overview".

Legal limitations are documented in the "Acrobat Reader Integration Key License Agreement".

Also, see Development of Plug-ins for Acrobat Reader in this FAQ.

### Using Acrobat Reader with the API

Both Acrobat and Acrobat Reader accept plug-ins. The difference between the two is that Reader can neither make changes to a file nor save a file. API methods that change a file in such a way that a save would be required are not available in Reader.

Note: Acrobat Reader only accepts "Reader-enabled" plug-ins. For more information, see Development of Plug-ins for Acrobat Reader. The "Acrobat Reader Integration Key License Agreement" also imposes additional restrictions.

### Using Acrobat Reader with the Interapplication Communication (IAC) API

Acrobat Reader is limited technically in what it can accomplish using IAC. Acrobat Reader was designed to view, print, and close a PDF file using IAC.

Acrobat Reader is not an OLE Server.

Acrobat Reader supports only the following DDE messages: FileOpen, FilePrint, FilePrintSilent, FilePrintTo, and AppExit.

Acrobat Reader supports only the four required Apple events: run, open, print, and quit.

For documented functionality as it pertains to using IAC with Acrobat Reader, see the "Acrobat Interapplication Communication Overview".

## Using Visual Basic with the Acrobat SDK

The Microsoft Windows versions of Acrobat and Acrobat Capture 2.0 software are OLE servers. The only requirement for using the OLE objects made available by Acrobat and Capture 2.0 is to have the product installed on your system and the appropriate `.tlb` file included in the project references for your Visual Basic project. The Acrobat `.tlb` file is named `Acrobat.tlb` and the Capture file is called `capserve.tlb`. These files are included in the IAC `headers.zip` file located on the Adobe public Web site.

Once you have the `.tlb` file included in your project, you will be able to use the Visual Basic object browser to browse the OLE objects. It is not sufficient to install just an ActiveX control or DLL to enable OLE Automation. You must have the full product (Acrobat or Capture 2.0) installed to use OLE automation.

### Getting Started

"Acrobat Interapplication Communication Overview" and "Acrobat Interapplication Communication Reference" document the objects and methods available. These documents (as well as the API) were designed with C programming in mind and programming with the API requires some familiarity with C concepts such as "enums."

The best resources for a VB programmer, besides the object browser, are the sample projects "VBOIW" and "VBDraw". These samples demonstrate use of the Acrobat OLE objects and contain comments describing the parameters for the more complicated methods.

## What Must I Know About C to Use Visual Basic with Acrobat?

Downloading the `IAC.h` (included in the IAC `headers.zip` file) will be helpful for a VB programmer. Some of the methods such as OpenInWindowEx can be confusing when utilized in VB. The method is prototyped as taking a `long` for the openflags parameter. The options for this parameter provided in the IAC Reference are:

AV_EXTERNAL_VIEW — Open the document with the toolbar visible.
AV_DOC_VIEW — Draw the page pane and scrollbars.
AV_PAGE_VIEW — Draw only the page pane.

If you were developing in the C Programming Language, these strings would be replaced by a numeric value prior to compilation; passing these strings to the method would not raise an error. When programming in VB, these strings are not replaced so you must look in the `IAC.h` header file to determine the appropriate numeric value. In `IAC.h`, you will find the following `enum`.

```
typedef enum _t_AVViewType {
AV_EXTERNAL_VIEW = 1, /* Open the document with toolbar visible */
    AV_DOC_VIEW = 2,       /* Draw the page pane and scrollbars */
    AV_PAGE_VIEW = 4       /* Draw only the page pane */
} AVViewType;
```

For example, if you wanted to draw only the PDF page, you would pass 4 for the openflags parameter.

In some situations, you will need to do a bitwise OR of multiple values and pass it to a method. An example of this is using PDDocSave. The ntype parameter of the PDDocSave method is a bitwise OR of the following flags as defined in `IAC.h`:

/* PDSaveFlags — used for PD-level Save
** All undefined flags should be set to zero.
** If either PDSaveCollectGarbage or PDSaveCopy are used, PDSaveFull must be used. */

```
typedef enum {
    PDSaveIncremental = 0x0000,  /* write changes only */
    PDSaveFull = 0x0001,         /* write entire file */
    PDSaveCopy = 0x0002,         /* write copy w/o affecting
      current state */
    PDSaveLinearized = 0x0004,   /* writes the file linearized */
    PDSaveCollectGarbage = 0x0020  /* perform garbage collection
      on unreferenced objects */
} PDSaveFlags;
```

If you want to fully save the PDF file and linearize it, you would pass 5 for the ntype parameter, since that is the binary OR of the PDSaveFull and PDSaveLinearized parameters. In many instances, the numeric values are spelled out in comments in the VB sample code, however, knowledge of why the methods are structured in this way and how the method would be utilized in C can help VB programmers when these values are not spelled out clearly.

## PDF Generation

There are currently five methods available for PDF creation. Three of these are supported by Acrobat Developer Support. The options are as follows:

### PDFWriter

PDFWriter emulates a printer driver and converts the GDI or QuickDraw commands from Windows and Macintosh applications to PDF documents. It is only available on the Windows and Macintosh platforms and is intended for desktop use by applications. For more information on PDFWriter, go to the Acrobat Main Page.

Note: PDFWriter is a client resource and can not be used as a server facility. Use of PDFWriter on a server or with simultaneous access from multiple applications or clients is unsupported and violates the terms of the End User License Agreement provided with the product. See What Is Supported And Unsupported in the Acrobat SDK? for more information. The PDFWriter API is documented in the "Acrobat PDFWriter API Reference", available from the documentation page on the Adobe public Web site and on the Acrobat 4.0 SDK CD.

### Acrobat Distiller

Acrobat Distiller is essentially a PostScript interpreter that can be used to convert PostScript to PDF. For more information on the Distiller program, go to the Acrobat Main Page. Distiller is the PDF creation application intended for batch processing use and for the creation of PDF files containing such high-end print publishing features as OPI comments, CMYK color spaces, and spot colors. Distiller also has the ability to interpret PostScript extensions called pdfmarks and convert them to PDF objects such as links, bookmarks, and annotations. See the "pdfmark Reference Manual" available from the documentation page on the Adobe public Web site and on the Acrobat 4.0 SDK CD. These high-end capabilities do not exist in PDFWriter.

Due to the built-in functionality of Distiller, you do not need to use the Distiller API to integrate it with your product. Distiller has the ability to watch directories over a network and convert any PostScript files saved to those directories to PDF. It is also possible to set different job options for each directory so that one directory can be used for high-end color output while the other can generate a more compressed file suitable for Web use. These features of Distiller are not supported by Acrobat Developer Support. Check the help documentation packaged with the product or books by Adobe Press and other publishers for using Acrobat Distiller through the user interface.

The Acrobat Distiller API can be used to programmatically process files and set the output path and file names. The API is documented in the "Acrobat Distiller API Reference" available from the documentation page on the Adobe public Web site and on the Acrobat 4.0 SDK CD.

## Writing PDF Files Using the Adobe PDF Library

The Adobe PDF Library is an object code library that exposes the PDFEdit API. This API is specifically designed for the creation and modification of PDF files. It is available on the Macintosh PowerPC, Windows 32-bit, and many UNIX platforms and can be linked in to your application to enable it to read, edit, and create PDF files. While the PDFEdit API makes PDF creation and modification easier than writing a PDF file from scratch, or parsing the page contents streams to modify the PDF file, it is not a "filter" that provides the simple creation of PDF files from other file formats.

Creation of PDF files with the Adobe PDF Library requires an understanding of the PDF file format as well as the PDFEdit API. An example of a typical use of the Adobe PDF Library is to convert a TIFF file to PDF. The Adobe PDF Library does not interpret the TIFF file and create a PDF file for you. The application that incorporates the library must be able to read the TIFF header to determine such attributes as color space, compression technique, bits per pixel, and so forth. After retrieving this information from the TIFF file, you need to strip the header information from the TIFF file to leave a simple RAW image file (only image data). The information retrieved from the TIFF file is then used to set up the graphics state prior to creating an image in the PDF file using the RAW image data. The sample application "addimage," available on the Adobe PDF Library SDK CD, demonstrates adding an image to a PDF file. The TIFF specification is available from the Adobe public Web site.

There are many sample applications available with the Adobe PDF Library that demonstrate different aspects of PDF generation and modification (see PDF Modification with PDFEdit). The "Adobe PDF Library Overview", which provides an introduction to using the Adobe PDF Library, is available from the documentation page on the Adobe public Web site and on the Acrobat 4.0 SDK CD.

The Adobe PDF Library is not free and is not available on the Adobe public Web site. For more information, see Licensing and Distribution Issues for Acrobat.

We are no longer licensing PDF Library 1.0.

## Converting Image Files to PDF Using Acrobat Capture 2.0

Acrobat Capture 2.0 is a Microsoft Windows OLE server Application that is designed for the batch conversion of image files to PDF and the recognition of the text in the resulting PDF files to create searchable text PDF files. For more information on Capture 2.0 software, go to the Acrobat Main Page. Due to the built-in functionality of Capture 2.0, many companies developing for in-house use do not need to use the Capture API to integrate it with their products. Capture 2.0 can watch directories and convert image files placed in those directories to PDF. Some developers, however, would like to customize the work flow and simplify the user interface. This development is possible through the Capture OLE Automation interface, as documented in the "Acrobat Capture API Reference" available from the documentation page on the Adobe public Web site and on the Acrobat 4.0 SDK CD. To perform such development, you need to purchase Acrobat Capture 2.0.

For a discussion on converting TIFF to PDF using the Adobe PDF Library, see Writing PDF Files Using the Adobe PDF Library.

### PDF File Generation From Your Application without Using Adobe Products

The PDF 1.3 specification is fully documented in the “Portable Document Format Reference Manual” available from the documentation page on the Adobe public Web site and on the Acrobat 4.0 SDK CD. Some developers have successfully developed the capability of generating PDF without the use of Adobe Products. Due to resource limitations, this development is unsupported by Acrobat Developer Support. The “Portable Document Format Reference Manual” is the best resource for this kind of development, as well as the use of Adobe products to create PDF files as a benchmark for your own development.

The PDF file format is quite complex, and developing code to generate it requires a significant amount of development. In estimating development time, we suggest you look at the methods used to compress and encrypt the page content streams, as well as the requirements of the byte offset table at the end of the PDF file. The LZW compression technique (used for Acrobat 2.1 compatible files) is the subject of United States patent number 4,558,302 owned by the Unisys Corporation. Independent software vendors may be required to license this patent directly from Unisys to develop software using LZW. Further information can be obtained from Welch Licensing Department, Law Department, M/S C2SW1, Unisys Corporation, Blue Bell, Pennsylvania, 19424.

## Modification of PDF

### PDF Modification with the API

Modification of the page contents of a PDF file is primarily accomplished through the use of the Acrobat API or the Adobe PDF Library. The OLE API to Acrobat is primarily limited to what a user can do through the user interface so while it is possible do such things as set DocInfo fields, create thumbnails, and add text notes, it is not possible to significantly modify the contents of a page. PDF Modification is not possible with the free Acrobat Reader.

The "AddElem" sample plug-in demonstrates how to add data to the contents of a page. Using the Acrobat API greatly simplifies modifying and creating PDF page contents. In addition, Acrobat updates the byte offset table and page resources to ensure that the PDF file is still readable after modification. If this were attempted by parsing and rewriting the page contents without using the Acrobat API, it would be significantly more difficult to do and could result in an unreadable PDF file.

### PDF Modification with PDFEdit

The PDFEdit API provides methods for reading a PDF file’s page contents and returning a display list that can be used to add and delete objects from the PDF page. The PDFEdit Library is discussed in the “Acrobat Core API Overview” and “Acrobat Core API Reference”, as well as the “Adobe PDF Library Overview”. These documents are available from the documentation page on the Adobe public Web site and on the Acrobat 4.0 SDK CD.

For more information on the Adobe PDF Library, see Adobe PDF Library.

## Text Extraction with the Acrobat SDK

The Adobe Acrobat Software Development Kit provides two ways for PDF to ASCII conversion. Through the Adobe Acrobat API, you can extract ASCII text from a PDF file. This requires the use of Acrobat and a plug-in developed in C or C++. The "Wordfind" sample plug-in demonstrates this functionality and can be used as a starting point for your own plug-in.

It is also possible to extract ASCII text from a PDF file using the Adobe PDF Library. The Adobe PDF Library contains a set of C object code libraries that can extract ASCII text without the use of Acrobat. Included with the libraries is sample code that demonstrates the text extraction capabilities and can be used as a starting point for your own application. For more information on extracting and highlighting text, see Extract and Highlight Text.

The Adobe PDF Library is not free and is not available on the Adobe public Web site. For more information, see Licensing and Distribution Issues for Acrobat.

We are no longer licensing PDF Library 1.0.

## Integration of Acrobat Viewers for PDF Display in an External Window

There are several ways to have the Acrobat program display a PDF file in an external application's window. You must have Acrobat installed on the system to view PDF in your own application's window.

There is no best way that we suggest to display PDF files in your application; you should examine the following list for the most appropriate method for your situation.

### Using Acrobat to View a PDF in Your Own Application's Window

Windows Only:

- OLE automation, using the OpenInWindowEx command for the 16-bit or 32-bit Windows Acrobat. This displays a live view of the PDF file in the OLE application window.

  For samples, see the "OIW" and "VBOIW" sample plug-ins.

- OLE automation, using the DrawEx command for the 16-bit or 32-bit Windows Acrobat. This displays a bitmap of the current page in the OLE application window.

  For samples, see the "Draw" and "VBDraw" sample plug-ins.

- Copy to Clipboard (New in Acrobat 3.01, available only on 32-bit systems). This copies a PDF image to the clipboard without requiring an hWnd or hDC using OLE Automation.

Mac OS Only:

- The "AEView" sample plug-in demonstrates rendering a PDF file into another application's window (see DrawIntoWindowCommand for details).

Windows, Mac OS, and UNIX:

- The "ExtrnWin" sample plug-in demonstrates displays a live view of the PDF file in a window created by the plug-in. You can extend this to display PDF files in an external application window.

### Using Acrobat Reader to Display a PDF in Your Own Application's Window

Due to restrictions in the "Acrobat Reader Integration Key License Agreement", you cannot write a plug-in to display PDF files in an external window.

- Mac OS and Windows: If your application provides the Netscape API, the Acrobat PDF Viewer plug-in provides support for rendering PDF files inside your application's window. This plug-in is part of the Acrobat and Acrobat Reader installations.
- Windows-only: The 32-bit version of Acrobat and Acrobat Reader provide an ActiveX Control designed for use with Microsoft Internet Explorer. Your application can also make use of this ActiveX Control for rendering PDF files inside of your application's window. Although technically feasible, this implementation is not supported.

Adobe does not currently support development using the Acrobat ActiveX (OCX) component that Acrobat ships with. Its purpose is to integrate with Internet Explorer. Adobe has no documentation available regarding ActiveX (OCX) development with Acrobat.

## What ActiveX Solutions Are Provided in the Acrobat SDK?

You may display a PDF file in your own application's window using the ActiveX Control provided with the Acrobat 32-bit Windows product. This displays a live view of the PDF file. The Acrobat 4.0 ActiveX Control only supports opening or printing PDF files. There is no documentation on the ActiveX in the Acrobat 4.0 SDK.

For more information on what development environments are supported with Acrobat, see Supported Development Environments.

## Multibyte PDF Document Handling

PDF files can have multibyte fonts. Acrobat 4.0 allows for the creation, modification, and use of multibyte PDF files. In multibyte documents, both text and other document information, such as bookmarks or text annotations, may be in multibyte format.

Several plug-in samples illustrate handling multibyte PDF documents. The API provides methods for handling multibyte documents.

For samples, see the "Templatex" and "Wordfex" sample plug-ins. You may use the Adobe PDF Library to extract multibyte text.

## Using Security with PDF

Adobe provides a 40-bit encryption scheme with Acrobat that queries the user for a password before a file is opened. The authorization procedure to open a file or to set an owner password can be modified by creating a custom security handler. Using a custom security handler, you could:

- Set up a secure CD-ROM where the PDF files can be opened without a password when they are on the CD and the plug-in is installed, but cannot be opened if they are moved to a system without the plug-in and the special file that existed on the CD.
- Set up a finger print scanner that determines if the user can open a PDF file and communicates with the plug-in to authorize the user to open the file.
- Set up a Web site that requires a user to have a particular IP address to open one of the files (that the plug-in would check before authorizing the file to be opened).

For more information on encryption in PDF, see the “Portable Document Format Reference Manual” and the “Acrobat Core API Overview”.

For samples of setting security, see the "Rot13" and "SetSec" sample plug-ins.

To programmatically enter a password and bypass the user interface prompt for password, you must replace the standard “security handler” for Acrobat with your own security handler. There is no way to bypass the password prompt with the standard Acrobat security handler. For an example of a custom security handler, see the "Rot13" sample plug-in.

## Using Full-Text-Search Systems with Acrobat and PDF

Adobe provides a full-text-search system for Acrobat 2.0 software and later versions based on the Verity search engine. However, this does not preclude other search systems from integrating with Acrobat.

The Acrobat search system was created using only public API and IAC calls, and you can easily remove it from Acrobat and replace it with another search technology. This section describes the tools available to full-text-search developers and the process for integrating full-text-search systems with Acrobat.

### Extract and Highlight Text

For indexing and searching PDF files directly, Acrobat provides support through IAC and plug-in calls.

For indexing PDF files, Acrobat provides text extraction APIs. Text extraction also supplies position information that can be used to highlight search hits in the original PDF file. The text extraction tools are provided as calls in the plug-in API on the Acrobat viewer platforms (Mac OS, Windows, and UNIX). A stand-alone text extraction Toolkit, a subset of the Adobe PDF Library, is available for Mac OS, Windows, and UNIX systems. These text extraction tools can be the basis of a PDF-only indexing product or can add support for PDF files to a current indexing product. This Toolkit is not free; see the Adobe PDF Library Page on the public Acrobat SDK Web site.

For further information on extracting text from PDF, see Text Extraction with the Acrobat SDK.

For samples, see the "Wordfind" and "Wordfex" sample plug-ins.

## Create a Search Plug-in for the Acrobat Viewer

The Acrobat Search plug-in that is provided with the Acrobat viewer is a true Acrobat plug-in and can be removed from the system simply by removing it from the Plug-ins directory and restarting Acrobat.

Full-text-search developers can create their own search plug-in by using the Acrobat API.

This search plug-in can be represented as one or more toolbar buttons or menu items, and they can use the capabilities provided in the API.

When creating a replacement search plug-in for the Acrobat viewer, you must decide what indices your search plug-in will use. You can choose to create your own indices (see Extract and Highlight Text) or to search the Verity indices created by the Acrobat Catalog product. To do the latter, you must license software directly from Verity. Verity can be contacted at 650-960-7600 or at http://www.verity.com.

The viewer interface can be extended with new menus and toolbar icons to allow tight integration with your search plug-in and Acrobat. For example, buttons to invoke a search and to find the next or previous occurrences can be added to the toolbar.

For samples of adding menu items and toolbar buttons, see the "Template" and "Templatex" sample plug-ins.

## Drive the Acrobat Search Plug-in

You can communicate with the Search plug-in via its plug-in API or via IAC (DDE or Apple events). Using either of these methods, you can control the Acrobat Search plug-in in the following ways:

- Control the list of indices to search

  The search interface allows for searches to be performed on one or more of the available indices. You can control the list of active indices.

- Initiate a search with options

  You can pass query expressions and search settings (for example, whole words only, word stemming, case sensitive, and so forth) to the Acrobat search plug-in and initiate the search. Search results will be presented to the user.

For samples, see the "HFTQuery" and "VBSrch" sample plug-ins.

For documentation, see the “Acrobat Search API Reference”.

### Creating Custom Doc Info Fields

1. You must create the custom DocInfo fields using either the Acrobat API or using Interapplication Communication (IAC). If you specify a non-existent DocInfo field with this method, it will be created. This method is documented in the "Acrobat Core API Reference" and the "Acrobat Interapplication Communication Reference".

2. Before indexing the PDF files with Acrobat Catalog, modify the `acrocat.ini` file so that the custom fields are properly indexed. This is explained in the online help files included with the Acrobat Catalog product.

3. Submit custom queries to the Acrobat Search plug-in to search by the custom fields, as demonstrated by the "HFTQuery" sample plug-in. The documentation for the search API is the "Acrobat Search API Reference". By specifying your custom DocInfo fields in the sortSpec parameter of the SearchExecuteQuery method, they will be used in the search.

## Customizing the Acrobat or Acrobat Reader Installer

There are two ways to customize the Acrobat install. One way is with the built-in InstallShield silent install option and the other way is with a Windows `.ini` file (`abcpy.ini`) that the installer checks during install. These methods allow for the setting of default install options and bypasses the display of dialog boxes. The latter allows you to display a progress monitor during install and displays the End User License Agreement, which is required by Adobe to be displayed.

The methods for modifying the installer are in the `abcpydoc.ini` file, included in the same directory as the installer executable on the multifile Reader installer CD (as opposed to the single file Reader installer that is downloaded from the Web site). It is also located on the Acrobat Publisher's Kit CD and the Acrobat product CD.

A few notes.

1. The installer will read the `abcpy.ini` file during install. Your installer can write out this file or add to it to control the install location.

2. The `.ini` file and InstallShield only work on the Windows platform.

### Visual Basic and the Registry

The following VB sample code launches Acrobat or Reader, based on its registry entry. This script extracts the registry entries generated by Acrobat to verify installation and then uses the specified path to launch the Acrobat Viewer. If both Acrobat and Reader are installed, only Acrobat is launched.

1. Copy and paste the code sample into VBProject. Some reformatting may be necessary.

```
Private Declare Function RegOpenKeyEx Lib "advapi32" Alias _
  "RegOpenKeyExA" (ByVal hKey As Long, ByVal lpSubKey As String, ByVal _
  dwReserved As Long, ByVal samDesired As Long, phkResult As Long) As Long

Private Declare Function RegQueryValueEx Lib "advapi32" Alias _
  "RegQueryValueExA" (ByVal hKey As Long, ByVal lpValueName$, ByVal _
  lpdwReserved As Long, lpdwType As Long, lpData As Any, lpcbData As _
  Long) As Long

Private Declare Function RegCloseKey Lib "advapi32" (ByVal hKey As Long) _
  As Long

Const HKEY_LOCAL_MACHINE As Long = &H80000002
```

```
'
'
'From FAQ the Acrobat Reader can be found at
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App
Paths\AcroRd32.exe"
'
'
Function RegGetString$(hInKey As Long, ByVal subkey$, ByVal valname$)
  Dim RetVal$, hSubKey As Long, dwType As Long, SZ As Long
  Dim R As Long
  RetVal$ = ""
  Const KEY_ALL_ACCESS As Long = &H3F
  Const ERROR_SUCCESS As Long = 0
  Const REG_SZ As Long = 1

  R = RegOpenKeyEx(hInKey, subkey$, 0, KEY_ALL_ACCESS, hSubKey)

  If R <> ERROR_SUCCESS Then GoTo Quit_Now

  SZ = 256: v$ = String$(SZ, 0)
  R = RegQueryValueEx(hSubKey, valname$, 0, dwType, ByVal v$, SZ)

  If R = ERROR_SUCCESS And dwType = REG_SZ Then
    SZ = SZ - 1
    RetVal$ = Left$(v$, SZ)
  Else
    RetVal$ = "--Not String--"
  End If

  If hInKey = 0 Then R = RegCloseKey(hSubKey)

  Quit_Now:
    RegGetString$ = RetVal$
End Function

Function Acrobat_viewer() As String
  Dim ViewerStr As String

  ViewerStr = RegGetString$(HKEY_LOCAL_MACHINE, _
    "SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\Acrobat.exe", _
    "Path")

  If ViewerStr <> "" Then
    ViewerStr = ViewerStr + "\Acrobat.exe"
    Debug.Print "Using Acrobat"
  Else
    ' Acrobat is not installed
    ViewerStr = RegGetString$(HKEY_LOCAL_MACHINE, _
      "SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\ _
      AcroRd32.exe", "Path")
    ViewerStr = ViewerStr + "\AcroRd32.exe"
    Debug.Print "Using Reader"
  End If

  Acrobat_viewer = ViewerStr

End Function

'
'
Private Sub Command1_Click()
  Unload Me
  End
```

```
End Sub

'
'
Private Sub Form_Load()
  Dim myRetVal As Integer
  Dim ShellStr As String

  ' Should really check to see if there is a viewer already running

  ' Get the path to the Reader from the registry
  ShellStr = Acrobat_viewer() '"C:\Program _
    Files\Acrobat3\Reader\AcroRd32.exe"

  ' Launch the application
  myRetVal = Shell(ShellStr, vbHide)

  ' Set up the appropriate DDE values and ask viewer to display a file.
  Text1.LinkMode = 0 ' End any previous sessions
  Text1.LinkTopic = "acroview|control" ' This is known
  Text1.LinkMode = 2
  Text1.LinkTimeout = 32767 ' Make the timeout suitable for any delay
  Text1.LinkExecute "[FileOpen(c:\acrobat3\exchange\license.pdf)]"
End Sub
```

2. Create a form with a text field and a button.
3. Verify that the path of the PDF file that will be launched is valid.

## Development of Plug-ins for Acrobat Reader

The Adobe Acrobat Reader Plug-in Program allows your plug-in, with certain restrictions (see license agreement), to run with the Acrobat Reader. An Adobe Acrobat Reader Integration Key License Agreement is required for this program. Download the PDF file of the agreement at http://partners.adobe.com/asn/developer/acrosdk/PDFS/readerlic.pdf or call our FaxYI fax response system at 650-556-8481 and request document #1234 to have a copy faxed back to you. Once you have the agreement, read and sign it, and then mail it to the ASN Developer Program to the address specified in the agreement with your license fee (faxed copies of the agreement will not be accepted).

For European Developers, you must be a registered developer in the European Adobe Solutions Network Developer Program. For more information, send e-mail to euroasn@adobe.com.

If you are thinking about developing a plug-in for Reader, keep the following in mind:

You may not develop a Reader plug-in that (the following list is an excerpt from the "Acrobat Reader Integration Key License Agreement"):

(i) Removes the menu item that calls up the Acrobat "About Screen" in Acrobat Reader;

(ii) Modifies or saves any file (including without limitation PDF (Portable Document Format), FDF (Forms Data Format), or annotation files) or enables another local or remote application or server to do the same; provided, however, that the restriction in this Section 2(a)(ii) shall not apply to Document-Independent Preferences which, for the purposes of this provision, means data used by the plug-in to set options, such as highlight color or default display font, that apply for all documents;

(iii) Opens encrypted documents without the authorized knowledge of the document passwords or violates the access rights specified for a document;

(iv) Displays a PDF document in the window of an application other than the Adobe Acrobat Reader;

(v) Accepts navigational commands from an application other than the Adobe Acrobat Reader;

(vi) Makes use of any function call from the Forms Host Function Table (HFT); or

(vii) Implements a Replacement File System for the Adobe Acrobat Reader.

For technical limitations, see the "Acrobat Core API Overview" and the "Acrobat Interapplication Communication Overview".

For instructions on how to make your plug-in Reader-enabled, see http://partners.adobe.com/asn/developer/acrosdk/main.html.

## Using Command Lines with Acrobat and Acrobat Reader under Windows

These are unsupported command lines, but have worked for some developers. There is no documentation for these commands other than what is listed below. You can display and print a PDF file using command lines with Acrobat and Acrobat Reader.

AcroRd32.exe filename — Executes the Reader and displays a file.

AcroRd32.exe /p filename — Executes the Reader and prints a file.

AcroRd32.exe /t path printername drivername portname — Initiates Acrobat Reader, prints a file while suppressing the Acrobat print dialog box, then terminates Reader.

The four parameters of the /t option evaluate to path, printername, drivername, and portname (all strings).

printername — The name of your printer.

drivername — Your printer driver's name. Whatever appears in the Driver Used box when you view your printer's properties.

portname — The printer's port. portname cannot contain any "/" characters; if it does, output is routed to the default port for that printer.

If using Acrobat, substitute `Acrobat.exe` in place of `AcroRd32.exe` in the command lines.

## Multithreading and Acrobat

No Acrobat products are multithreaded at this time and any attempts to access the Acrobat API in a multithreaded manner will cause the viewer to hang or crash. Acrobat and the Adobe PDF Library API make methods available that can help a plug-in or application manage its thread's access to the Acrobat API. These methods involve registering for notification of an event.

This is accomplished with the following methods, as documented in the "Acrobat Core API Reference".

```
AVAppRegisterNotification
AVAppRegisterIdleProc
```

The Adobe PDF Library has separate methods for registering for notifications, as documented in the "Acrobat Core API Reference". When Registering for a notification, the method is passed a function to be called by the Acrobat viewers or the Adobe PDF Library when the event occurs. Registering for an IdleProc calls the function when nothing else is occurring. All notifications and IdleProcs are queued and called in order. Registering for notifications of events or IdleProcs can help a multithreaded application ensure that Acrobat is not accessed by multiple threads simultaneously; however, it is still the application's responsibility to manage its threads that access Acrobat.

## Using Acrobat Forms and the FDF Toolkit

The Acrobat Forms Author Plug-in can be used to create forms fields in PDF files. It is available for Apple Macintosh and Microsoft Windows platforms. It requires Acrobat 3.01 and later. The Acrobat Forms Fill-in Plug-in, used to fill in and submit PDF files, is available for Apple Macintosh, Microsoft Windows platforms, and also most UNIX environments. It requires Acrobat Reader 3.01 and later. Both Acrobat Forms plug-ins and other important resources, such as example Javascripts, sample forms, and numerous tips and tricks, are available from the Adobe Web site at http://www.adobe.com/prodindex/acrobat/formsresources.html

A user with the Adobe Acrobat Reader and the Forms Fill-in Plug-in or Acrobat and the Acrobat Forms Author Plug-in can fill in and submit forms to a server. It is also possible to submit the data to a "mailto:" address to put the FDF data in an e-mail. Once the user fills out the PDF-based form, the data can be submitted to the server for processing. Data can be submitted from a PDF form in either FDF (Forms Data Format) or HTML. Data can only be imported into a PDF form if it is in FDF format. The FDF format is a simple text file that has a structure based on the PDF file format. It is discussed in the "Portable Document Format Reference Manual".

In general, the Forms plug-ins are designed to be run in a Web environment. Every FDF file contains a reference to a PDF file that the data is intended for, designated with the "/F" key inside the FDF file (unless the FDF file is for the same Form that you submitted from). When Acrobat or Acrobat Reader (with the Forms plug-in) is sent an FDF file, it opens the appropriate PDF file, and fills the form fields with the data from the FDF file. If the PDF file is referenced by a URL (for example, http://yourcompany.com/file.pdf), the FDF file must be sent in response to a submit action from a PDF form.

A plug-in to Acrobat can programmatically import FDF data into a PDF file from a local file system using the HFT made available by the Forms plug-in. For more information, see the "Acrobat Forms API Reference" and the "Acrobat Core API Overview". Acrobat 4.0 lets you use (OLE) Automation to programatically add/modify/delete form fields, import/export FDF, execute Javascripts, and much more. For more information, see the "Acrobat Forms API Reference".

If you need to parse FDF files submitted from a PDF form, or generate FDF files to be submitted to a PDF form, you can use the FDF Toolkit. The FDF Toolkit is an object code library available on the Macintosh PowerPC, UNIX, or Microsoft Windows 32-bit platforms. There is a C/C++/Java/Perl FDF Toolkit for Windows, Mac OS, and UNIX. There is also an ActiveX FDF Toolkit for Windows. For detailed information on creating and parsing FDF files, see the documentation included with the FDF Toolkit. To download the FDF Toolkit and access the FDF Toolkit documentation, see the Acrobat Forms Data Format Toolkit.

## Using the FDF Toolkit and PDF Templates

Starting with version 3.5 of the Forms plug-ins, the capability to dynamically construct PDF files from Template pages is available. The Acrobat Forms Author Plug-in lets you define a page in your document as a Template, which can then be used to generate, or spawn, new duplicate PDF pages on-the-fly. For more information on working with Templates, see the help documentation included with the plug-ins.

An issue that can be confusing for developers working with FDF is that an FDF file cannot open a PDF and dynamically add a page to it. An FDF file can either dynamically assemble a PDF document from Templates and fill in the form fields, or, open a PDF file and fill in the form fields, but cannot do both. The difference is subtle, yet significant. For this reason using the methods FDFSetFile and FDFAddTemplate, when constructing a single FDF file, will create an invalid FDF file.

To create an FDF file that dynamically creates a PDF file from Templates using the FDF Toolkit involves calling FDFAddTemplate for the first page followed by FDFSetValue (or any other FDFSet method) to specify the data to be filled into the form fields for that page. Calling FDFAddTemplate again creates a new page that can be setup like the previous page.

When passing field names to methods such as FDFSetValue or any other method taking a field name, use the original field name from the Template even if you passed true for bRename in FDFAddTemplate. Conceptually the fields are renamed after the data has populated the PDFTemplate and the field names will not match up if you use the renamed field name when creating the FDF.

## Configuration Tips and Tricks

1. If you are submitting from an Acrobat Form and the server returns FDF, then the URL must end in "# FDF". For example:

   `<http://myserver/cgi-bin/myscript# FDF>`

2. The URL may be relative (to the URL of the Form that you are submitting from).

3. You may set the SubmitForm URL to a “mailto:” scheme.

4. If your script is producing FDF, then you need to ensure that it emits the correct MIME type, that is, `application/vnd.fdf`.

5. If your server returns a static FDF file, as opposed to one dynamically generated by a script, then you may have to define `application/vnd.fdf` as a new MIME type on your server.

6. The FDF returned from the server does not require an "/F" key if the FDF is for the same Form that you submitted from.

7. It does require an "/F" key (if you are using the FdfTk, then use FDFSetFile) if the FDF is for a different Form than the one you submitted from.

8. The value of the "/F" key may be relative (to the URL of the Form that you submitted from).

9. If you are submitting from an HTML form, then the URL does not need to end in "# FDF".
10. The returned FDF must include an "/F" key giving the absolute URL of the PDF that it is for.
11. The PDF will automatically get loaded by Acrobat upon opening the FDF.

#### Browser Caveats

When you submit from HTML and return FDF (which points to the PDF):

- Under Internet Explorer, the PDF will open in a new browser window.
- Under Netscape 4.0x, it opens in the same window that you submitted from, and even in the same frame (if you are using frames).

#### How Javascript Events are Processed

As you enter keystrokes, you get keystroke events. Once you hit Enter, or tab or click outside of a field, you get a validate event. If the field is a number, then you get a calculate event and all fields in the form that have calculations get recomputed (only once) in the order given by the Calculation Order (Tools->Forms->Set Field Calculation Order). Finally, all fields that have changed as a result of the calculations and which have formatting scripts attached, are executed. This only changes the appearance of each field, but not the value (for example, if you entered "1/1/98" as a date, that is the value, but it may get formatted for display purposes only as "January 1, 1998").

### Useful (But Not So Obvious) FDF Toolkit Tips

#### SetValue

- For radio buttons and checkboxes, pass `true` for the last parameter.
- The `newValue` parameter must be either `Off` (to uncheck the checkbox), or a value that was entered as the “Export Value” when defining the properties of the field in Acrobat (to check the checkbox or radio button).

#### SetStatus

- You can use this call to cause an alert to pop up at the client.
- You can use a text field to display a message.
- You can create a “status” field in your form that has no border and no background color, and is read-only, and it will be invisible until some text is written to it via FDF.

#### SetFile

- Use this API to have the FDF point to the PDF that it is for.
- Use it only if the Acrobat Form from which the submit action took place is not the one that the FDF is for.
- This call is mutually exclusive with `AddTemplate`.
- You can use a URL that is relative to the Acrobat Form from which the submit action took place.
- If the submit occurred from an HTML form, you must use an absolute URL.

SetOpt

- If you use this API (to change the options for a listbox or combobox), make sure you also call SetValue.

SetFlags

Here are some useful idioms:

| | |
|---|---|
| Hide the field | SetFlags(< field name>, FDFSetF, 2) |
| Show the field | SetFlags(< field name>, FDFClrF, 2) |
| Make the field read-only | SetFlags(< field name>, FDFSetFf, 1) |
| Make the field writable | SetFlags(< field name>, FDFClearFf, 1) |

FDFSetAP

- Acrobat will only import an /AP from FDF into fields of type “Button.”
- Only the FDFNormalAP will be imported (FDFDownAP and FDFRolloverAP will be ignored) unless the button that the /AP is being imported into has a “Highlight” of type “Push.”
- Be aware that the new imported /AP will not show if the “Layout” for the button is of type “Text only.”
- If the picture looks too small inside the button field with too much white space around it, once the FDF containing the new /AP is imported into the Acrobat Form, you may want to crop (using Acrobat) the PDF page used as the source of the /AP.

Set< some action> Action

- Reprogram the form on the fly. Use these calls to dynamically change the action associated with a button. For example, SetSubmitFormAction, SetResetFormAction, and SetJavaScriptAction.

## Templates Tips

1. Templates is an Acrobat-only feature: it does not work in the free Reader.
2. An FDF can either cause an existing PDF to be opened using FDFSetFile or create a new PDF by spawning pages from templates contained in PDF documents specified by the FDF. Do not use FDFSetFile, FDFSetID, or FDFSetStatus.
3. Structure your FDF-generating program like this:

```
FDFCreate
<begin loop>
  FDFAddTemplate
  FDFSetValue (or FDFSetFlags,
  FDFSetOpt, etc.)
<end loop>
FDFSaveToFile
FDFClose
```

4. FDFAddTemplate: Pass the URL of the PDF containing the template as parameter.

   - You can use URLs that are relative to the Acrobat Form from which the submit action took place.
   - You may pass an empty string as the URL, in which case the template is expected to reside (possibly as a hidden template) inside the Acrobat Form from which the submit action took place.

5. FDFSetValue (or FDFSetFlags, FDFSetOpt, and so forth.) Remember to use the field names as they appear in the template and not the field names after the templates are spawned (as they will be renamed).

   - The purpose of renaming fields when spawning pages is to avoid conflicts when one template is spawned multiple times onto the same PDF, or when unrelated fields in various templates have the same name.
   - Fields get renamed to `P< page number>.<template name>_ <template number on this page>.<field name>`. For example, `P3.flowers_0.rose`.

6. If you are choosing to rename fields and you are using Javascript, make sure you use the right field names. For example:

```
var cFldName = "Description";
var cMe = event. target. name;
if (cMe. substring( 0, 3) == "P1.") {
  cFldName = "P1.OrderForm_ 0." +
  cFldName ;
}
```

7. You can spawn multiple templates onto the same page (that is, overlays).

# Links to Documentation

## Acrobat Product Information

Acrobat Main Page

## Acrobat Software Developers Kit

Acrobat SDK Main Page

## SDK Overview

“Release Notes for Adobe Acrobat 4.05 SDK”

“Getting Started Using the Adobe Acrobat Software Development Kit (SDK)”

“Adobe Acrobat Software Development Kit Contents Description”

“Samples Roadmap”

“Code Sample Descriptions”

## File Formats

“Portable Document Format Reference Manual”

“Highlight File Format”

## Acrobat Core API

“Acrobat Core API Overview”

“Acrobat Core API Reference”

“Acrobat API Development”

## Other Acrobat Viewer Plug-ins

“Acrobat Digital Signature API Reference”

“Acrobat Forms API Reference”

“Acrobat Search API Reference”

“Acrobat Weblink API Reference”

“Other Plug-ins”

## Interapplication Communication

“Acrobat Interapplication Communication Overview”

“Acrobat Interapplication Communication Reference”

### Acrobat Distiller Application

"pdfmark Reference Manual"

"Acrobat Distiller API Reference"

"Acrobat Distiller Parameters"

### Other Acrobat Products

"Acrobat Capture API Reference"

"Acrobat Catalog API Reference"

"Acrobat PDFWriter API Reference"

### Other Related Documentation

"Acrobat Reader Integration Key License Agreement"

"Adobe PDF Library Overview"

"Adobe PDF Library License Agreement"

## Acrobat Software Developer Resources

### Acrobat Developer Resource Links

Acrobat Resources Page

PDF NewsGroups

Information on Partner Resources

PDFZone

BCL-Computers

### Adobe Solutions Network Developer Program Information

ASN Developer Program Home Page

### Acrobat Reader

Distributing Acrobat Reader

Adobe Acrobat Reader Plug-in Program

### Acrobat Toolkits/Libraries

Adobe PDF Library

Acrobat Forms Data Format Toolkit

“Forms Data Format Toolkit Reference”

### Customer Support

Adobe Customer Support Page

## Samples (Pre-compiled code)

### Windows IAC Samples

"CaptOLE" Demonstrates how to communicate through OLE with the Acrobat Capture 2.0 product.

"VBCapOLE" demonstrates how to communicate through OLE with the Acrobat Capture 2.0 product.

"DDEOpen" demonstrates a basic DDE connection to the Acrobat viewer.

"Distill" demonstrates communication with the Distiller program.

"Draw" illustrates how to render the PDF page contents into another application’s window using OLE 2.0 with Visual C++.

"VBDraw" illustrates how to render the PDF page contents into another application’s window using OLE 2.0 with Visual Basic.

"FormsOLE" demonstrates how to create and modified Form fields using Forms OLE Automation.

"OIW" illustrates how to display a PDF file within another application’s window using OLE 2.0 with Visual C++.

"VBOIW" illustrates how to display a PDF file within another application’s window using OLE 2.0 with Visual Basic.

"PdfwCtrl" demonstrates using the Windows API Escape function with PDFWriter. Pdfwctrl opens files and coverts them to PDF using PDFWriter. To use this sample, select the PDFWriter as the default printer.

"VBSrch" demonstrates how to communicate with Acrobat Search using functions exported by a DLL.

### Macintosh IAC Samples

"AEView" demonstrates how to control various components of Acrobat for the Macintosh using Interapplication Communication and C programming.

"OpenAll-Dist" demonstrates how to remotely control the Acrobat Distiller using AppleScript.

"SCRIPTS" AppleScripts that demonstrate various operations.

## Plug-in API Samples

W, M denotes platform compatibility.

"AddElem" demonstrates how to modify the page contents of a PDF file (W, M)

"AddPS" modifies the viewer’s print stream. (W, M)

"Balloon" creates a new action type for links. When a link is created it can be given a new action of type “balloon help”. (W, M)

"DDEClnt" demonstrates how to communicate with an Acrobat plug-in from an external application using DDE. (W, M)

"DebugWin" demonstrates how a plug-in can export its own HFT. (W)

"ExtrnWin" demonstrates how to open an AVDoc in a developer-created window. (W, M)

"FormDemo" demonstrates how to import and use the AcroForms plug-in HFT. Shows resetting fields, enumerating fields, and exporting data. (W,M)

"HFTQuery" demonstrates how to communicate with the Search plug-in through the Search plug-in’s HFT. Shows how to set the word options and number of documents, add and remove indices, and initiate a search query. (W, M)

"ImageSel" implements a selection server, which allows images to be selected on the page. (W, M)

"Notify" keeps track of event notifications in the viewer. (W, M)

"Progbar" demonstrates how to create and use a custom progress monitor. (W, M)

"RFS" demonstrates how to replace a file system. (W, M)

"Rot13" demonstrates how to create a simple custom security handler. The new security handler takes a password and rotates all of the letters by 13 characters and stores that in the file. (W, M)

"RplcDemo" replaces the AVAlertNote method, appends the string “rplcdemo was here” to the passed string parameter, then sends this new string out via the original AVAlertNote method. (W, M)

"SeeAnnot" demonstrates how to create a tool that recognizes only a particular type of annotation. SeeAnnot ignores all annotations except ones created by the Stamper plug-in. For these, the cursor changes to the Stamper shape over a Stamper annotation; if the annotation is clicked on, a dialog box is displayed. (W, M)

"SetSec" demonstrates setting PDF file security data programmatically using the “Standard” security handler. The Preferences item allows setting passwords and security option defaults. (W, M)

"SnapZoom" creates a new tool, SnapZoom, which allows you to zoom in or out on a page when the mouse button is pressed. (W, M)

"Stamper" implements a rubber stamp tool. It illustrates a custom tool and a custom annotation. (W, M)

"Starter" Skeleton for a plug-in. (W, M)

"Template" Plug-in template, which follows plug-in UI style guidelines. (W, M)

"WebLHFT" demonstrates how to register a new Weblink driver with the Weblink plug-in. (W)

"Wordfind" demonstrates how to extract text from a PDF file and highlight text in a PDF file. it also shows how to highlight text based on character offsets instead of word offsets. (W, M)

## Japanese Samples

"Templtex" Essentially the same as Template, but demonstrates adding a double-byte menu item, adding a Unicode or host-encoded bookmark, and getting the string from a Unicode or host-encoded bookmark. (W, M)

"Wordfex" Essentially the same as WordFind, but demonstrates extracting the words from a PDF file that contains multibyte text. Text can be extracted either as a Unicode string or as a host-encoded string. (W, M)