May 2005

.NET Developers Series

How-To-Select™ Guides

How-To-Select a PDF Component for .NET™

Covers Adobe Portable Document Format components for use in both WinForm and WebForm/ASP.NET applications

By Mike Gunderloy

Categories discussed in this Guide:

- PDF Viewers
- PDF Generators
- XML-to-PDF Processors
- PDF Print Drivers
- PDF Editors
- PDF Forms Products
- PDF Converters

http://www.howtoselectguides.com/dotnet/pdf/

Table of Contents

Introduction

Welcome to Xtras' How-To-Select™ Guides

Hello and welcome to the first issue of Xtras' How-To-Select™ Guides: How-To-Select a PDF Component for .NET™. We write our .NET Developer Series Guides for people who develop software designed to run on Microsoft's .NET platform, whether they program in C#, VB.NET, or another .NET language.

In this specific Guide edition we tackle PDF components designed to be used by .NET developers when creating applications with Visual Studio .NET or other similar tools. We cover scenarios where you might consider using a PDF product, the various functionality you can expect from available products, decision points you might have when you consider using a PDF product, and almost all of the vendors and products in the space.

Our goal with the How-To-Select Guides series, however, is to provide you with clear, correct, unbiased information to help you choose the .NET components and tools you need to make your own projects a success. The Guides are published by Xtras, a company that has been in business since 1994 (originally as VBxtras), serving the developer community as a source of information about and reseller of components and tools for Microsoft-centric developers and as part of our goal we want to make our process as transparent as possible.

To ensure both transparency and the highest quality possible, Xtras selected Mike Gunderloy to be the Executive Editor of the .NET Developer Series, and the author of this first Guide. Mike is also the editor of Larkware.com, and a well-known author and developer in the .NET community. Writers for the series are chosen jointly by the Publisher and the Executive Editor.

If you questions, comments, or suggestions, please let us know at http://forums.howtoselectguides.com/dotnet/pdf/ or email us at feedback@howtoselectguides.com. We hope you like what we have in store. Read on!

-The How-To-Select Guides Team

Editor's Note

Dear .NET Developer:

Have you ever tried to find a component for your .NET application only to waste more time in research than you saved in using the component? Or gotten 80% of the way to launch, only to discover that the component you were depending on lacked one crucial feature? Depending on your project's scope, such problems can range from minor annoyances to major disasters. But with all the great .NET tools and components out there on the market, these problems can be avoided...if only you can find the right components for your own particular needs.

Unfortunately, getting information on components has been a hit-or-miss proposition. That's why we're launching the How-To-SelectTM Guide series. Our goal is simple: we want to bring you the comprehensive, unbiased information that you need to make an informed choice when picking the components that you need to make your .NET project a success. Each How-To-Select Guide will look at a particular category of components in depth. We'll offer guidance on what components in that category can do, and show you how to choose between the various components on the market. We'll also list and evaluate every component that we can in the category (limited only by the vendor's willingness to provide us with their software to evaluate).

How-To-Select Guides go far beyond simple feature checklists to address the issues that really matter to working developers who need to find the right components for serious projects. We'll help you sort out the use cases for the various components, discuss vendor support and licensing policies, and make sure you understand what each component can (and can't) do for project. We include free and open source software right alongside commercial offerings, so you can choose the most economical component that will get the job done for you.

We're also committed to keeping the Guides up-to-date.

Assuming changes and ongoing interest in each category, we plan to revise each Guide roughly every six months with the latest products and version information, as well as additions and corrections based on reader feedback. If you've got something to say about our coverage, we'd love to hear from you on our forums here via our web site's forum. Why not stop by to let us know how we can help your .NET development efforts succeed?

Mike Gunderloy
Executive Editor and Author
How-To-Select™ Guides - .NET Developer Series

Publisher's Note

Dear .NET Developer:

Hello and welcome to the very first How-To-Select™ Guide from Xtras, Inc. We put forth a tremendous effort to launch the How-To-Select Guides, and I hope you are as excited about what they offer .NET developers as we are.

First we helped you FIND Components and Tools

Back in 1994 I was both awed by the explosion of third party components and tools for Visual Basic and overwhelmed when trying to find the ones I needed and "Googling" just wasn't an option back then. So I launched the VBxtras [www.vbxtras.com] catalog of components and tools with a goal of helping developers find components and tools for Visual Basic. Then in 2002 I launched the Xtras.Net [www.xtras.net] website helping developers find components and tools for .NET.

Now we are Helping you SELECT

I had a passionate vision, bordering on obsessive, to to more; to provide detailed enough information that developers could select components and tools. But it wasn't clear how to accomplish this until Mike Gunderloy agreed to help us launch our How-To-Select as Executive Editor of the .NET Developer Series.

We focus on Categoies, not Products

You'll find product reviews elsewhere, but not here. Our How-To-Select Guides focus on the product categories with product information in context helping you quickly become a category expert and ensuring they have a long shelf life. We also plan do revise each guide biannually to keep up with changes in the category.

Our Goals for the How-To-Select Guides

We set forth several goals for our How-To-Select Guides. We wanted them to be:

• **Clear & Concise** - Quick to read and easy to comprehend.
• **Accurate & Unbiased** - Completely Defensible among leading experts with as little bias as humanly possible.
• **Exhaustive, Thorough, & Complete** - Covering all options include commercial, shareware, freeware, open source, and even coding techniques; all decision points and aspects of potential concern; and including all currently available products in the category.

They are Living Guides

I think you'll agree we've come very close to achieving our goals. But we are only human and there are aspects we forgot, ones we didn't realize were important, and ones we probably omitting. But that's okay because our Guides live on at http://forums.howtoselectguides.com. There you'll find lively discussions about this and many other categories. Visit often with your questions about selecting components or tools for .NET

Help us create your Ultimate Component & Tools Selection Resource

We know our How-To-Select Guides can save you time and help you avoid costly mistakes but we need your help to make them even better:

• **Subscribe** - Subscriptions are free, so sign up on our website to receive each Guide as it's released.
• **Tell Others** - If you know .NET developers, send them a link to http://www.howtoselectguides.com/ and suggest they subscribe too.
• **Mention in Newsgroups** - When you answer questions in developer newsgroups and forums about components and tools, mention and include a link to our Guides.
• **Blog about a Guide** - If you blog and are inspired by our Guides, mention and include a link to that Guide.
• **Ask Questions** - If our guides don't answer a question you have, ask it in the forum so one of our authors and your fellow .NET developers can answer the question for you.
• **Answer Questions** - Answer questions asked in our forum by of other .NET developers, which also helps others with the same question.
• **Make Suggestions** - If you notice anything we can improve about our Guides, post a suggestion in our forums.
• **Offer to Write** - If you are a category expert, or want to become one, and you are a good writer willing to meet deadlines, offer to write one of our future Guides.

• **Encourage Vendor Participation** - Tell vendors being included in relevant Guides is a prerequisite for considering purchase of their products.
• **Support our Sponsors** - Please consider supporting the companies that sponsored and/or advertised in these Guides, such as TallComponents and Xtras.Net in this Guide edition.

Thanks for reading about our new How-To-Select Guides, and I hope this and future Guides will both save you time and make sure you make the right selection!
Oh, and don't forget to subscribe!

Sincerely,
Mike Schinkel
Publisher
How-To-Select™ Guides

Overview of Adobe PDF

The Adobe Portable Document Format (PDF)

Adobe Acrobat files (often simply called PDF files, from "Portable Document Format") are certainly one of the great success stories of personal computing. Adobe's product strategy combined two key features. First, they developed an open file format that could handle complex document layouts including illustrations and typography with excellent fidelity. Second, they gave away reader software across a wide variety of operating systems. The result has been to make PDF a de facto standard for distributing high-quality documents of all sorts, from instructional manuals to crossword puzzles.

The ubiquity of PDF files makes them very attractive for many purposes. When you want to generate attractive typeset content that will be accessible to viewers on many platforms, PDF files are a natural choice. Adobe sells applications such as Adobe Acrobat 7.0 Professional that allow you to create PDF files from Word documents, scanned images, and other sources. Other vendors, too, have entered the market with their own programs that can produce PDF output. But such manual solutions do not fill the needs of .NET application developers who are looking for integrated PDF functionality.

In this How-To-Select™ Guide, we'll look at components that are designed to help you work with PDF files directly from your .NET applications. These components cover a wide range of functionality, from simply viewing existing PDF files to creating entirely new PDF files from scratch to more specialized areas such as PDF forms. We'll cover both commercial and opern-source offerings, and help you understand the wide range of components available in this area.

The PDF Specification

Adobe publishes a Web-based reference to the PDF specification. You can download a copy from the Adobe Web site. The current version of the PDF Specification is the Fifth Edition, which covers version 1.6 of the specification, and which was issued in November 2004. Naturally, it's published in PDF form itself. If you want to understand all of the capabilities of PDF files, from annotations to zooming, it's a must-have. The specification also contains all of the information that you need to build your own PDF files from scratch by stringing together the internal typesetting codes that make up a PDF file. Most developers, though, will find it much easier to purchase a component to do this work instead!

Adobe Reader Versions

As it has added new features to the Acrobat specification, Adobe has issued new versions of the free Adobe Reader application. For this How-To-Select™ Guide, we tested using the most recent release, Adobe Reader 7.0. Just about every PDF document in current use should also display fine in the previous generation software, Adobe Reader 6.0.

Functionality by Categories

No one piece of software will do everything you can possibly think of with PDF files. The format itself is flexible enough to be used in many different ways, and component vendors have come up with a variety of tools and components that you can use in your own PDF applications. In this Guide we've collected examples of seven different kinds of PDF software:

- **PDF Viewers**
- **PDF Generators**
- **XML-to-PDF Processors**
- **PDF Print Drivers**
- **PDF Editord**
- **PDF Forms Products**
- **PDF Converters**

It's important to realize that these categories are not mutually exclusive. Some products, for example, can generate and edit PDF files, as well as manipulate PDF forms.

PDF Viewers

Sometimes you just want to be able to view existing PDF files. In this case, you need to find software that can act as a PDF viewer. This might be a pre-built application that you can distribute in place of Acrobat Reader, a Windows Forms control that you can drop into your own application, or a class library that can render PDF content to a bitmap or other raster image.

PDF Generators

When you need to create a new PDF document "from scratch," then you should consider a PDF generator. Software in this category often provides an object model for PDF files that allows you to build up documents from smaller entities such as pages, paragraphs, tables, and images. Other products get their instructions from XML files while others are implemented as printer drivers; we discuss these latter to in their own sectrion.

When choosing a object model based PDF generator, consider the richness of the object model and the amount of control it offers over placement and formatting of individual elements within the PDF.

XML-to-PDF Processors

XML processors are a special case of PDF generators. An XML processor reads an XML file containing formatted instructions and from it generates PDF. There are two flavors of XML processors: ones that process XML documents marked up with proprietary XML schemas and ones that process XML documents marked up using the industry-standard XSL-FO technology. XSL-FO is a page-oriented markup language that is part of the W3C Extensible Stylesheet Language Recommendation. One advantage of using XSL-FO rather than a proprietary XML dialect is that many XSL-FO processors can render their output to other formats in addition to PDF.

An XML processor can take an XML document in the proper schema and generate output in PDF (or in some cases other printable formats). Such a processing engine is useful when you're generating your PDFs from XML output created by other applications. XML processors are also ideal if you are treating your PDF generation as part of a standard-based document generation process.

PDF Print Drivers

One common strategy for creating PDF documents is to simply defer the entire problem to the Windows printing subsystem. That is, instead of equipping your application with any specific knowledge of the PDF format, you enable your application's users to print documents to any Windows printer. Then you install a special printer driver that, instead of printing to a physical printer, directs the printed output to a PDF file instead.

This method allows you to use the same methods to develop a PDF document that you would any other printed document (for example, the methods of the System.Drawing and System.Drawing.Printing namespaces). The end user of your application then selects the PDF printer when they wish to create a PDF document.

In most cases, such virtual printers are installed on a system-wide basis, so that they become available from any application that the user has installed. Some vendors also provide PDF printer drivers as libraries that you can call specifically from your own applications without installing them as a system printer.

PDF Editors

In addition to creating new PDF documents from scratch, you may wish to modify existing PDF documents. In this case, you'll need to find a component that includes PDF editing capabilities. Here you'll need to investigate carefully to match your needs to the capabilities of a particular product.
Some components can do little more than tack additional pages on to the end of an existing file. Others can insert new pages at arbitrary locations, add new content to an existing page, split or merge PDF documents, or extract content from a PDF document to a plain text file. A high-end editing product can even give you full programmatic access to change the existing content in a PDF file.
Note that we're using the term "editor" in a somewhat narrower sense than you'll sometimes encounter in discussing software. While a text editor or an image editor ordinarily let you both create new documents and manipulate new documents, most these PDF editors are targeted specifically at editing tasks rather than document creation.

PDF Forms Products

If you want to work with PDF interactive forms (also known as AcroForms) you'll need a forms-capable component. Here you'll find a variety of different levels of support, depending on whether you want to create PDF files containing form fields, automatically fill in form fields in code, or submit filled-in forms using form actions.

PDF Converters
Finally, you may wish to convert PDF content to some other format that's easier to work with. PDF viewers can be seen as a special case of this category, converting PDF files to bitmaps. Another example is the conversion of PDF files to SVG files hosted on HTML pages for Web delivery of PDF documents.

Related Categories
There are developer products that can generate and/or otherwise work with PDF even though the primary focus of those those products are not PDF. We'll be covering these related categories in future HowTo-Select™ Guides:

• Reporting Products
• Graphing and Charting Products
• Imaging Products

If one of these categories interests you greatly, let us know via our online forums at http://forums.howtoselectguides.com.

Reporting Products
Many reporting packages can export their finished reports as PDF files. These packages are the most appropriate choice if your primary interest is in databound reports that depend directly on ADO.NET data sources. While you can use any object-oriented PDF generator to create a banded report based on hierarchical data, you'll find it much easier to do so with a true reporting package.

Graphing and Charting Products
Several graphing and charting products can export their finished graphics to a variety of file formats including PDF. If your primary interest in the PDF file format is to have a high-fidelity way to reproduce graphs and charts, using a charting product may be a better bet than trying to build up the chart yourself from the graphics primitives provided by a PDF generator.

Imaging Products
Some imaging products provide general-purpose functionality to convert between PDF files and various graphics file formats. While these tools do not have any granular understanding of the internal structure of PDF files, they may be useful if your main goal is to publish PDF files to the Web or display them on a form, or to create PDF files from existing graphic-files.

Scenarios
There are many different ways that you might want to use PDF files in your own software, for example:

• Embedded PDF Viewers
• Structured PDF Generation
• Unstructured PDF Generation
• Dynamic PDF Creation
• Databound PDF Report Creation
• Generating PDF Output From CMS
• HTML and PDF Form Merging

While it's impossible to explore every possibility, ths section of the Guide offers some general guidance to common scenarios.

Embedded Viewer
You may wish to allow users of your application to open and view arbitrary PDF files. In this case, you should consider using an embedded, self-contained PDF viewer component. A control is often the simplest way to go here, though a class library may offer you more flexibility.

Structured PDF Generation
You may need to build PDF documents that have a relatively fixed structure. For example, you might have an accounting application that needs to generate invoices. In this case, a batch-mode PDF generator that can create PDF from a text file or a markup file, or a PDF print driver, may serve your needs with the least effort.

Unstructured PDF Generation
If your application allows users to generate PDF files with an arbitrary structure, perhaps by drawing or typing content, might need a more flexible generation method. In this case, one of the PDF generators that implements a full object model for PDF files is likely to be your best bet.

Dynamic Document Creation
Sometimes you'll want to create dynamic PDF documents based on user selections - for example, you might want to assemble a custom product catalog from a range of possible catalog pages. In this case, it's often effective to have all possible pages pre-built and to use an editing product that can merge pages into a larger document to produce the final PDF files.

Databound Report Creation

In many cases you may want to produce a PDF report directly based on data stored in a database. In this case, you might want to consider using a full-fledged databound reporting product instead of the type of product we review in this particular Guide. However, depending on your needs, you might still be better off with a PDF-focused product some of them do support data-binding via ADO.NET.
We'll be covering reporting products in a future How-To-Select™ Guide.

CMS Document Output

If you're integrating PDF document creation with a content management system, it's likely that you are working with an existing XML-based infrastructure. In this case, you should investigate XML processors as likely being the best fit to your existing processes, especially if your CMS already uses XSL-FO.

HTML and PDF Form Merging

It may be convenient to let people fill out forms as HTML and then provide them with a finished product as high-fidelity PDF in their browsers. You can accomplish this with a suitable PDF forms product that lets you edit a basic PDF document with form fields and "flatten" the result into a static PDF document that you then output to the user's browser.

TALL™ components
Finally, a .NET PDF component that's easier to work with than most people
THE .NET PLATFORM DESERVES MORE THAN WRAPPERS :-)

Our 100% **.NET PDF creation component** offers state-of-the-art simplicity. A pull-model calls back to your code – whenever it needs content – to complete a page. Allowing you to dynamically create PDF documents with the highest speed, the lowest memory consumption and **extreme scalability**.

We also provide tried and trusted 100% .NET components to **split, stitch, stamp, query** and **form-fill** encrypted PDF documents. And they're so simple anyone could use them, well almost.

Visit our website for live demos, online documentation and to download fully functional evaluation copies.
www.tallcomponents.com

FEATURING: DYNAMIC PDF / NO THIRD-PARTY DEPENDENCY / 100% MANAGED / FULLY UNICODE / RIGHT-TO-LEFT TEXT / PASSWORD PROTECTION GENERATE FROM XML OR PROGRAMMATICALLY / GDI+ SUPPORT / EMBED ALL RASTER IMAGES / EMBED TRUETYPE FONTS / HYPERLINKS / CHARTS AND TABLES HEADERS AND FOOTERS / FORM-FILL / STAMP / SPLIT / APPEND / STITCH / ENCRYPT

Decision Points

With all the different software in the PDF category, it can be difficult to sort out exactly what you need to make your application a success. In this section of the Guide, we'll offer advice about factors that should influence your decision.

Naturally, not all of these factors will apply to every application.

Try Before You Buy

The first thing to keep in mind is that no matter how much information we pack into this Guide, there's no substitute for trying the software in your own environment. No one knows your own requirements, or the quirks of your own tools and components, better than your do. Fortunately, just about every vendor in the .NET tools and components business provides a trial version of their software for free evaluation these days. We recommend that you use this Guide to locate the components that seem like a good fit for your project, and then download copies to further evaluate them in your own environment. With PDF components, you'll find that evaluation copies normally print some sort of banner or watermark on the generated or displayed document. This is not enough to interfere with evaluation, but generally precludes you from using an evaluation copy in a commercial application. Of course, you should follow the licensing requirements scrupulously in any case.

Consider Vendor's Other Products

In many cases, the PDF software that we're reviewing is only part of a vendor's product line. Some vendors provide a much broader line of developer components and tools. Other vendors sell complementary end-user applications.

If a single vendor can supply multiple components that you need, you may benefit from reduced licensing fees by buying the components together in the form of a suite. This can also cut your learning curve by giving you products that follow similar conventions and that work well together. In the case of end-user applications, you might find that you need to do less work to complete your application because your end users can use these vendor-supplied pieces instead of custom-developed functionality.

Choosing a PDF Viewer

In many cases, you won't need to choose a PDF viewer at all, because you can depend on the end user to download a copy of the free Acrobat Reader and use that. But there may be times when you need or want an embedded solution; i.e. where the viewer is an integral part of your own application. In this case, you'll need to evaluate the various controls and class libraries available. A control may provide the simplest interface, but generally the class libraries provide more flexibility. For the easiest integration with the .NET Framework, you will probably want to consider one that interfaces to the existing Graphics class. Viewers that use their own proprietary drawing libraries instead are likely to require more code to implement in your application.

Choosing a PDF Generator

If your goals include building PDF files from scratch, you'll want to look at one of the many PDF generator products on the market. You'll quickly find that these products span a variety of price points and capabilities. Understanding your choices will help you narrow down the selection to the products that will best fit your needs.

Generation Strategies

Generally speaking, you can choose between seven different methods of programmatically generating PDF files:

Object Model Synthesis - This is the most common approach. Starting with some top-level object, such as Document or Pdf, you build up your PDF by adding objects representing text, images, bookmarks, and the other parts of a PDF file. Within this category, you'll find three broad approaches to the problem of representing the structure of a PDF file. Some products place a very thin wrapper over the low-level structure used by Adobe for PDF itself. Others start from the object model developed in the System.Drawing namespace of the .NET Framework, and mimic it to produce a set of objects and members that are more familiar to .NET developers. Still others develop their own abstract model of the objects that make up a document. When you're using an object model to build PDF files, you need to decide which of these styles you prefer. You also need to investigate the richness of the object model to determine whether it includes objects representing all of the features that you'd like to use in your document. For example, if you want to include a table, do you need to draw it with lines and calculate where to place text yourself, or is there some higher-level abstraction representing a table already baked into the object model?

Windows Forms Control - Instead of providing an abstract class, some vendors provide an object model wrapped as a Windows Forms control or an ActiveX control that can be instantiated on a Windows Form. This approach tends to be similar to the object model approach as far as building the actual document, but precludes you from using with ASP.NET, windowless Windows Services, and/or console applications.

Event-Driven Generation - This approach is analogous to the method used by the System.Drawing.Printing namespace in the .NET Framework. With this approach, you create a top-level object and register your own event handlers for events such as the start of printing each page. When your code is called back in response to these events, it's up to you to supply the appropriate objects to be printed. This sort of "pull" approach can help minimize memory requirements for large documents.

Streaming Generation - Another approach that can result in lower memory requirements for large documents is an API that allows streaming a PDF directly to a FileStream or other output object. Typically, products that implement such an API let you open a file and then send paragraphs of text, images, and other objects to the file without maintaining the entire document in memory. While this way of producing a PDF can lower the resource hit for producing large documents, it also typically constrains you from using some features such as cross-references within a document.

Merging Data - Some products can generate PDF files by merging data from existing files such as text and image files. These may be plain text files, or files containing special markup codes that are interpreted by the batch generation software. This approach is best suited for applications such as batch generation of invoices or form letters.

XML Transformation - Several products implement their own XML Document Object Model to represent the parts of a PDF. With these products, you can directly open an XML document that follows the correct DOM and immediately save it as PDF with no further processing. Alternatively, you can use XSL to transform another XML document to the expected format and then save it as PDF. You may also be able to combine this sort of XML-based PDF creation with object model based modification in a hybrid approach.

XSL-FO Processing - Another way to combine PDF generation with XML standards is to use an XSL-FO formatting engine together with the XSL-FO standard. Several of the products in this Guide implement XSL-FO engines in .NET code. See the section "Understanding XSL-FO" later in this Guide for more information.

Output Methods

If you're generating PDF, you should consider where you'll be storing the generated PDFs. Various products offer more or less flexibility in their output APIs.

Disk Files - At the most basic level, any product should be able to generate a file on disk. But you may need more flexibility than that.

Stream Object - Some products let you send their output to any Stream object (FileStream, MemoryStream, and so on).

ASP.NET Response Object - If you're working in an ASP.NET application, you will probably want a product that can write directly to the ASP.NET Response object. The alternative is to output PDFs to temporary files and then point the user's Web browser to those temporary files. But if you do that, you have to manage the process of cleaning up the temporary files after some arbitrary time period, at which point the URLs to your previously-generated PDFs will break.

Special Features

Finally, you should pay attention to the special features that may make your life easier. Any PDF generator will let you place text and lines on a page, and build up a document from multiple pages. But the PDF specification encompasses much more sophistication than that, and various products also bring in their own notions of more sophisticated objects. We can't cover every facet of every generator, but here are some representative capabilities that you might find useful:

Font Embedding - The PDF specification allows for embedding TrueType and PostScript fonts directly in a document, so that the document will look right even on a computer that doesn't have the font installed. More sophisticated versions allow embedding partial fonts (only the characters that you actually use) and Unicode fonts.

Images - If you're working with images, check to make sure that the format you want to use is supported by the product you're considering.

Graphics - Support for drawing graphics using lines, arcs, circles, and so on varies widely. Some products offer only a few methods in this area; others have rich object models with scaling, rotation, transparency, and the ability to construct your own composite objects.

Tables - The PDF specification itself doesn't deal with tables, but many PDF generators include methods designed specifically to build tables.

Text flow - If you're composing documents from large amounts of text, you may want a product that can automatically handle flowing text from one paragraph, table cell, or page to another.

Annotations - PDF annotations can be notes, highlighting, or even embedded files. If you need these features, you'll find that this limits the software that you can use to generate PDFs.

Bookmarks - PDF bookmarks are used to generate the hierarchical outline that serves as a table of contents in Acrobat Reader. For long documents, you'll want a product that can easily insert bookmarks into your documents.
Hyperlinks - The PDF specification allows hyperlinks both within a document and to external URLs. If you want to make use of this facility, make sure you pick a product that understands hyper-linking.

Understanding XSL-FO

If you're looking for a standards-based approach to building PDF files, you should investigate XSL-FO. The "XSL" stands for "eXtensible Style Language" and the FO stands for "Formatting Objects." XSL-FO is a part of the W3C Extensible Stylesheet Language (XSL) Version 1.0 Recommendation and it provides a standard set of XSL tags for transforming XML documents into a page-oriented print representation, with headers, footers, text paragraphs, images, columns, tables, and so on.

XSL-FO is not tied explicitly to PDF; an XSL-FO processor can read an XSL-FO file and generate print-ready output. Depending on the particular XSL-FO processor, this output may be in formats such as PostScript or RTF in addition to PDF. This gives you a certain amount of freedom. If you store the source for your PDF files in XML format and transform into XSL-FO you can later choose a different physical output format by supplying different parameters to the XSL-FO formatting engine. Whether it's actually practical to do so will depend on your use of any vendor-specific extensions to the XSL-FO standard, availability of an XSL-FO formatter for the output format you need, licensing fees, and (if you find yourself needing to swich engines) compatibility. At the time of this writing few XSL-FO processors for .NET or otherwise support the entire XSL-FO specification, and in many cases those that claim support don't implement everything 100% correctly.

Of course if you like the idea of storing your data in XML and using XSL transforms but don't expect a benefit from using W3C standard XSL-FO, you can select one of the products that use a proprietary XML schema to generate your PDFs.

Understanding PDF Encryption

The PDF standard includes encryption and password features to help secure documents. You can encrypt a PDF document, which makes it useless hash except when decrypted. To decrypt the document, you must supply one of two passwords, the user password or the owner password. The user password allows you to open and view the document, but may not allow full control of the document. When a document is encrypted, you can specify that some operations, such as printing, copying text, or modifying the document, should not be allowed if only the user password is supplied. The owner password allows you to do anything with the document, including changing encryption settings. If you need to work with encrypted documents, you'll need to be sure that you purchase a product that understands PDF encryption and that allows you to supply appropriate passwords when necessary.

Four Levels of PDF Editing

"Editing" is a term that covers a wide variety of capabilities in this class of software. Roughly speaking, you can distinguish four levels of PDF editing, in increasing order of power and complexity.

If your plans include editing PDF files, it's important to define just what level of editing capability you need, and to confirm that the products you're evaluating include the capabilities that you require.

The four levels of editing PDFs include:

Rearranging Existing Content - The simplest level of editing is concerned with rearranging existing content. Typically this includes merging PDF files together, splitting a PDF file into individual pages, and inserting or appending pages. More advanced capabilities include rotating and scaling pages and placing multiple "thumbnails" on a single page.

Add New Content - The next step up is the ability to add new content to an existing PDF file without altering the existing content. This might include stamping watermarks (text or graphics) on to existing pages, adding bookmarks, or even plasing new text or graphics in an overlay layer.

Modify Existing Content - A complete editing product will also provide you with the means to read and alter existing page content. Typically this includes an API for retrieving text and graphics objects from a page and changing their properties.

Access Low-Level API - Finally, for the ultimate in control, you may gain access to the low-level API defined by Adobe in the PDF specification. Access at this level means that you can make any change to a PDF file that is supported by the PDF format itself.

Features

The following list of features is provided by category and then by specific feature.

General Features

These are features that apply broadly across many types of software.

Managed Code - The product is 100% managed code, as opposed to unmanaged code or a managed code wrapper around unmanaged code.

Integrated Help - The product provides a help file integrated directly with the Visual Studio .NET help file.

External Help - The product provides an external help file.

Documentation - The product provides documentation beyond a help file such as a manual or code samples.

Peer Support - There are peer support options such as newsgroups or discussion boards available for the product.

Vendor support - The vendor provides direct support options for the product, either as part of the purchase price or as a separate support contract.

Compact Framework support - The product includes a version that works with the .NET Compact Framework

PDF Generation

Generate Text - Place text in an arbitrary font at an arbitrary position on the page.

Embed TrueType Fonts - Embed TrueType Fonts in the generated PDF document.

Embed Type 1 Fonts - Embed PostScript Type 1 Fonts in the generated PDF document.

Embed Images - Embed Images in the generated PDF document.

Create Graphics - Draw simple graphics such as lines, circles, and rectangles.

Create Annotations - Add annotations to the generated PDF document.

Create Tables - Create tables in the generated PDF document by supporting an object model of rows and columns, rather than requiring you to work with low-level line-drawing and cell-by-cell text placement.

Page Markers - Create page markers such as "page m of n"

Persistent Content - Create content that repeats across every page in a document, such as a header, footer, or watermark.

Section Content - Create content that repeats across every page in a section, such as a header, footer, or watermark that appears on only some pages of a document.

Bookmarks and Outlines - Create a hierarchy of bookmarks to locations within the document that can be displayed to form a document outline in the Acrobat Reader.

Layout Grid - Create a grid on the page to help with object positioning. This is useful during the page design process to locate other page elements.

Background Image - Create an image that overlays an entire page for use as a watermark.

Hyperlink - Create a hyperlink to another location in the document or to a Web site.

Transforms - Shift, rotate, and scale elements on the page.

Transparency - Adjust the transparency of elements on the page.

Bar Codes - Create bar codes in various formats such as UPC and Code 39.

Embed Attachments - The PDF specification allows embedding file attachments of any type in a PDF file. The user can open an attachment directly from the PDF file, or save it to disk for later examination.

Generate from XML - Provides an XML schema that can be directly transformed to PDF with a single operation.

Streaming Generation - Generates PDF in a streaming fashion to keep memory use low for large files. This may be implemented by responding to events, similar to the .NET printing model, or as an API that streams content directly to an open file. "Streaming" is used in the conceptual sense here; such products may or may not make use of .NET Stream objects.

XSL-FO - Implements the XSL-FO standard for generating PDF files from XSL-FO files.

PDF Editing

Alter Existing Content - Alter existing content in an existing PDF file.

Add New Pages - Add pages to an existing PDF file.

Add New Content - Add new content to an existing page in an existing PDF file.

Merge Files - Merge two PDF files into a single PDF file.

Split Files - Split a single PDF file into multiple files.

Import Pages - Import pages from one PDF file into another PDF file

Transform Pages - Rotate, scale, or clip imported pages when placing in a new PDF file.

PDF Forms

Create Form Fields - Place form fields on a newly-generated PDF page.

Fill Out Form Fields - Supply content for form fields in an existing PDF file.

Flatten Forms - Convert form fields into static content

Form Actions - Trigger form actions such as submitting a form to a URL.

PDF Viewing

Standalone Viewer - Includes a pre-built standalone viewer application that can be used as an alternative to Acrobat Reader.

Viewer Control - Includes a Windows Forms control that can display a PDF file.

Viewer Class - Includes an abstract viewer class that can be used to convert a PDF file to another format for viewing.

Vendors

Amyuni Technologies
Vendor/Author: Amyuni Technologies
Website: http://www.amyuni.com
Address: 1255, boulevard Laird, Suite 101, Mont-Royal, Québec, H3P 2T1, CANADA

Aspose Pty Ltd
Vendor/Author: Aspose Pty Ltd
Website: http://www.aspose.com
Address: 41, Lily Street, Hurstville, NSW, 2220, AUSTRALIA

ceTe Software
Vendor/Author: ceTe Software
Website: http://www.cete.com/
Address: 7815 Old Georgetown Road, Suite 200, Bethesda, MD 20814, USA

ComponentOne LLC
Vendor/Author: ComponentOne LLC
Website: http://www.componentone.com
Address: 4516 Henry Street, Suite 500, Pittsburgh, PA 15213, USA

FyTek
Vendor/Author: FyTek
Website: http://www.fytek.com/
Address: 29200 Vassar, Suite 540, Livonia, MI 48152, USA

Gnostice Information Technologies Private Limited
Vendor/Author: Gnostice Information Technologies Private Limited
Website: http://www.gnostice.com
Address: #45, Floor - I, Sankey Road, Palace Orchards, Bangalore 560 003, INDIA

Gerald Henson
Vendor/Author: Gerald Henson

Kazuya Ujihara
Vendor/Author: Kazuya Ujihara

O2 Solutions
Vendor/Author: O2 Solutions
Website: http://www.o2sol.com/
Address: 18/6 Eftimie Murgu St., 3400, Cluj-Napoca, ROMANIA

PDFTron Systems
Vendor/Author: PDFTron Systems
Website: http://www.pdftron.com/
Address: 2575 West 4th Ave, Unit 303, Vancouver, B.C., V6K 1P5, CANADA

Pegasus Imaging Corporation
Vendor/Author: Pegasus Imaging Corporation
Website: http://www.pegasusimaging.com
Address: 4522 West Spruce Street, Suite 200, Tampa, FL 33607, USA

root-software ag
Vendor/Author: root-software ag
Address: Bèrglen, SWITZERLAND

Andrea Canegrati
Vendor/Author: Andrea Canegrati
Website: http://sharppdf.it/Home.asp

Syncfusion Inc.
Vendor/Author: Syncfusion Inc.
Website: http://www.syncfusion.com
Address: 9001, Aerial Center Parkway, Suite 110, Morrisville NC 27560, USA

TallComponents BV
Vendor/Author: TallComponents BV
Website: http://www.tallcomponents.com/
Address: Venloseweg 7a, 5931 GR Tegelen, THE NETHER-LANDS

Visual Programming Ltd.
Vendor/Author: Visual Programming Ltd.
Website: http://www.xmlpdf.com/index.html
Address: PO Box 22 222, Khandallah, Wellington, NEW ZEALAND

Products Covered in this Guide

- **Amyuni Technologies**
  - Amyuni PDF Creator Developer Version - v2.0
  - Amyuni PDF Converter .NET Developer Version - v2.5

- **Aspose Pty Ltd**
  - Aspose.PDF - v2.1.5.0
  - Aspose.Pdf.Fo - v1.1
  - Aspose.Pdf.Form - v1.0.4
  - Aspose.Pdf.Kit - v1.0.3

- **ceTe Software**
  - DynamicPDF Generator for .NET - v3.0.3
  - DynamicPDF Merger for .NET - v3.0.3

- **ComponentOne LLC**
  - ComponentOne PDF for .NET - v1.1

- **FyTek**
  - FyTek's PDF Forms - v3.1 Windows EXE, DLL, .NET Version
  - FyTek's PDF Meld - v5.2 Windows EXE, DLL, .NET Version
  - FyTek's PDF Report zWriter - v3.2 .NET Version
  - FyTek's Text2PDF - v3.1 .NET Version

- **Gnostice Information Technologies Private Limited**
  - eDocEngine ActiveX/.NET - v2.0
  - PDFtoolkit ActiveX/.NET - v2.0

- **Gerald Henson**
  - iTextSharp - v0.04

- **Kazuya Ujihara**
  - iText.NET - v1.2.1-1

- **O2 Solutions**
  - PDF4NET - v2.6

- **PDFTron Systems**
  - PDFTron PDF PageMaster - v1.1
  - PDFTron PDF2SVG - v1.1
  - PDFNet SDK - v2.12

- **Pegasus Imaging Corporation**
  - PDFXpress Professional - v1.0

- **root-software ag**
  - Report.NET - v0.08.01

- **Andrea Canegrati**
  - SharpPDF - v2.0

- **Syncfusion Inc.**
  - Essential PDF - v3.0.1.0

- **TallComponents BV**
  - PDFKit.NET - v1.0
  - PDFRasterizer.NET - v1.0
  - TallPDF.NET - v2.0

- **Visual Programming Ltd.**
  - Visual Programming Ibex PDF Creator - v3.0

Amyuni PDF Creator Developer Version

Version 2.0, starting at $920
www.amyuni.com/en/products/pdf_creator/demo.html
Licensing: Proprietary, per developer

Amyuni PDF Creator comes in both end-user and developer versions. The end user version is a PDF workbench application thatlets you open PDF files, create new files, edit, annotate, and print them, and fill out PDF forms. It also allows import and export in various other formats including RTF, Excel, HTML, and JPG. The Developer version brings these same capabilities to a .NET control that you can use directly in a Windows Forms application, or to a library that you can call in code. Thus you can use it either from the user interface of your application or simply behind the scenes as a PDF creation and manipulation library.

PDF Creator supports creating most aspects of PDF files, including form fields, annotations, bookmarks and hyperlinks, and graphics. The library supports encrypting and security PDF files. You can create an entire document in memory, but you can also flush pages to a file one at a time, freeing the memory for each one, to aid in creating large documents. On the editing side, you can add content to existing pages, add, insert, delete, and move pages, and edit existing content with reflow.

Documentation is supplied in the form of a PDF file. The company offers technical support via e-mail with extended support available for purchase, as well as through an online forum. There is a trial version of PDF Creator available for download. Pricing is per application, for an unlimited number of developers and full redistribution.

Amyuni PDF Converter .NET Developer Version

Version 2.5, starting at $800
www.amyuni.com/en/products/pdf_converter/demo.html
Licensing: Proprietary, per developer

Amyuni PDF Converter comes in several versions. The end-user version is a printer driver that allows creating a PDF document from any Windows application that can print. The developer version has the same basic functionality, but it is licensed for royalty-free distribution and built so that it can only be used from within your own application. When your application is running, the printer (to which you can assign any name you choose) is active; otherwise, as far as the user's system is concerned, it doesn't exist. Redistribution is a simple matter of copying the right files and running an activation procedure in your application's initialization code.

Documentation is supplied in the form of a PDF file, which covers all versions of PDF Converter (there are also ActiveX and Win32 DLL entry points to the developer version). The company offers technical support via e-mail with extended support available for purchase, as well as through an online forum. There is a trial version of PDF Creator available for download. Pricing covers an unlimited number of applications and developers with redistribution rights.

Aspose.PDF

Version 2.1.5.0, Starting at $329
www.aspose.com/Products/Aspose.Pdf/Downloads.html
Licensing: Proprietary

Aspose.PDF provides PDF creation through three methods. First, there's a very fine-grained object model. Second, you can transform an XML file directly into a PDF file with only a few lines of code. Third, there's a streaming model that allows you to open a PDF file and write directly to it, one paragraph at a time. This last mode is ideal for keeping memory requirements low on large documents, though you lose some features because it is by its nature a forward-only API.

Aspose.PDF's object model offers a document broken up into sections, which each contain a collection of paragraphs, as well as such objects as OddHeader, OddFooter, EvenHeader, EvenFooter, and so on. The Paragraph itself is an abstract class; you'll actually add specializations of the Paragraph such as Text, Image, Table, or Graph (which represents a simple graphic, not a graph in the mathematical sense). You generally create a document by creating objects, setting their properties, and then adding them to the appropriate collections. MSDN-style help covers all of this. There's also a Programmer's Guide in the form of online help, but this is actually online, hosted on the Aspose Web site; you'll probably want to use a Web crawler application to pull a copy to your local development machine if you decide to use this library. There's a set of samples provided in either C# or VB .NET as an ASP.NET application that shows how to mimic the reports in the Northwind sample database.

Other features supported by the object-oriented API include PDF encryption, generating form fields (though not filling them out), annotations and file attachments, and automatic generation of tables of contents and lists of tables and figures.

You can also use Aspose.PDF to create PDF documents directly from XML documents. These documents can either be structured as XML in the precise schema that Aspose.PDF uses, or transformed to that schema with XSL. This is as simple as calling the BindXML method to bind an instance of the Pdf object to an XML file, and then calling the Save method to save it back out as PDF. You can save PDF to a file, a stream, or the HTTP Response object in an ASP.NET application.

Aspose.PDF also supports a unique "direct-to-file" mode for creating large PDF documents with minimal memory usage. In this mode, you open a Pdf object directly on a filestream, append a section to it, and start adding paragraphs to the section. Each paragraph is flushed directly to the filestream as it is written. You'll lose some functionality (for example, the $P macro for the total page number obviously can't work if the total number of pages isn't known) but if you're generating large documents this may be a worthwhile tradeoff.

Aspose offers a variety of licensing plans depending on the number of developers you intend to have using the product and your deployment process. Depending on the plan, you can end up with licenses for a single or multiple developers, internal deployment, deployment to a single customer site, or OEM deployment to multiple customer sites. There are also slight feature differences between Professional and Enterprise licenses. Support is free and via public forums or e-mail. You can download an evaluation copy of Aspose.PDF from the Aspose Web site.

Aspose.Pdf.Fo

Version 1.1, Starting at $99
www.aspose.com/Products/Aspose.Pdf.Fo/Downloads.html
Licensing: Proprietary

Aspose.Pdf.Fo is an XSL-FO engine written in 100% C#. It apparently implements most of the XSL 1.0 version of XSL-FO, though Aspose has not yet published detailed conformance information. Transparency and SVG images are not supported in this version. To use the software, you supply an XSL-FO file or appropriate XML and XSL files, or a stream or XmlDocument containing the appropriate information, to the Converter object. You then call the Save method to get back an output Stream or file with the PDF output. The software installs with MSDN-style help on the Converter class; further help is available in a rather sketchy Programmer's Guide on the Aspose Web site.

Aspose offers a variety of licensing plans depending on the number of developers you intend to have using the product and your deployment process. Depending on the plan, you can end up with licenses for a single or multiple developers, internal deployment, deployment to a single customer site, or OEM deployment to multiple customer sites. Support is free and via public forums or e-mail. You can download an evaluation copy of Aspose.Pdf.Fo from the Aspose Web site.

Aspose.Pdf.Form

Version 1.0.4, Starting at $119
www.aspose.com/Products/Aspose.Pdf.Form/Downloads.html
Licensing: Proprietary

Aspose.Pdf.Form provides a simple programmatic interface to existing PDF forms. You can use it to enumerate the fields in a document, to get the values of current form fields, or to assign new values to existing fields. After changing the value of fields you can also save the filled-in document. The software installs with MSDN-style help on the Form class; further help is available in a short Programmer's Guide on the Aspose Web site.

Aspose offers a variety of licensing plans depending on the number of developers you intend to have using the product and your deployment process. Depending on the plan, you can end up with licenses for a single or multiple developers, internal deployment, deployment to a single customer site, or OEM deployment to multiple customer sites. Support is free and via public forums or e-mail. You can download an evaluation copy of Aspose.Pdf.Form from the Aspose Web site.

Aspose.Pdf.Kit

Version 1.0.3, Starting at
www.aspose.com/Products/Aspose.Pdf.Kit/Downloads.html
Licensing: Proprietary

Aspose.Pdf.Kit is a product that's difficult to categorize. It's an editor for PDF files, but rather than offer general-purpose editing, it bundles together a set of very specific functions. If you need these particular functions, you can get them through a clean, object-oriented API. For anything else, you'll need to look elsewhere:

• Combining and splitting PDF files, appending pages, extracting pages, and inserting pages
• Reading or setting the PDF document properties such as author and title.
• Adding a logo or watermark to every page of a PDF file.
• Encrypting or decrypting a PDF file.

Documentation is supplied in MSDN-style help. There is some additional information in a Programmer's Guide on the company's Web site.

Aspose offers a variety of licensing plans depending on the number of developers you intend to have using the product and your deployment process. Depending on the plan, you can end up with licenses for a single or multiple developers, internal deployment, deployment to a single customer site, or OEM deployment to multiple customer sites. Support is free and via public forums or e-mail. You can download an evaluation copy of Aspose.Pdf.Kit from the Aspose Web site.

DynamicPDF Generator for .NET

Version 3.0.3, starting at $199
www.cete.com/Products/GeneratorForNET/Download.csp
Licensing: Proprietary; can be licensed per-server or per-developer with redistribution rights

DynamicPDF Generator for .NET is a class library that's designed to generate PDF files directly to disk, to any System.IO.Stream object, or straight to the IIS output stream in an ASP.NET application. It comes in three editions: a free Community Edition with a minimal set of page elements, and Professional and Enterprise Editions that support an increasing number of more complex elements. You can evaluate the entire object model and all of the page elements using the Community Edition; if you use one of the higher-end elements without the corresponding license, DynamicPDF puts its own watermark on the page.

Creating a PDF document with DynamicPDF Generator is an iterative process using a reasonably simple object model. First you create a Document object to represent the entire document. Then you create a Page object to represent the first page of the document. Then you create individual PageElement objects such as Label, Link, Note, or Rectangle objects. These are the objects that have properties indicating their color, placement, text, and so on.

Add the PageElement objects to the Page, then add the Page to the Document. Repeat until you have all the pages that you need, and then output the document (which is trivially simple; you just call the Draw or DrawToWeb method). Everything is placed using standard typographic points (72 points to the inch).

There's good support for flowing and formatting text here. The TextArea and HtmlTextArea page elements both support text continuation with a method that lets you retrieve the overflow:

```
MyTextArea = MyTextArea.GetOverflowTextArea()
```

In addition, the HtmlTextArea page element lets you use HTML tags (as well as some non-HTML attributes) for formatting. Tables are supported through a typical object model of rows, columns, and cells. You can place anything you want into a table cell, and control the overflow from one cell to another. DynamicPDF Generator also supports PDF security with both 40 and 128 bit encryption.

The software is 100% managed code, strong-named and ready to deploy to the GAC if that's your preference though it is not deployed that way by default. It's also suitable for XCOPY deployment, since it has no external dependencies other than the .NET Framework. It's tested on the 1.0, 1.1, and 2.0 beta Frameworks.

The software ships with a whole batch of samples in both C# and VB.NET: 17 Web Form examples, 2 Windows Forms examples, 1 console application, and 2 examples showing how to create custom page elements. Custom page elements inherit from the PageElement base class and override its Draw method to let you output anything you want to the PDF document. You also get MSDN-style help that provides both an overview of the library's operation and complete details on its members. The product documentation is also online for browsing at the ceTe Web site. Standard support includes 24-hour e-mail response, and you can purchase priority support including telephone support and e-mail support with a 2-hour response time.

DynamicPDF Merger for .NET

Version 3.0.3, starting at $399
www.cete.com/Products/MergerForNET/Download.csp
Licensing: Proprietary; can be licensed per-server or per-developer with redistribution rights

DynamicPDF Merger is a superset of ceTe's DynamicPDF Generator product; depending on which license of Merger you buy, you automatically get a corresponding Generator license to go with it. What Merger brings to the table, in addition to the ability to generate PDF files, is editing and merging existing PDF files, as well as basic PDF forms functionality.

On the merging side of things, you can merge documents together, append documents to the current document, or pluck out single pages and add them to a document. You can also create new pages using the full object model from DynamicPDF Generator and merge them into the current document. Merged pages can be cropped, rotatated, or scaled, which allows you to (among other things) build a single new page out of multiple existing pages. You can also add new content to existing pages. Merger also allows you to read and modify PDF form fields, as well as to flatten fields into static content. You have access to the raw binary contents of imported pages if you want to muck around with them.

As with Generator, Merger ships with many samples and MSDN-style help that's integrated with the Visual Studio .NET help collection. The product documentation is also online for browsing at the ceTe Web site. Standard support includes 24-hour e-mail response, and you can purchase priority support including telephone support and e-mail support with a 2-hour response time.

ComponentOne PDF for .NET

Version 1.1, starting at $299.95
www.componentone.com
Licensing: Proprietary, per-developer with redistribution rights

ComponentOne PDF for .NET offers an approach to building PDF files that should feel very natural to anyone who has worked extensively with the .NET Framework. Rather than design an object model that is a thin wrapper around the low-level structure of the PDF format, ComponentOne modeled their C1.C1Pdf namespace on the System.Drawing namespace. If you're already familiar with using the Graphics object to display text and graphics in .NET, you'll find the transition to ComponentOne PDF for building PDF documents an easy one. You use the same Font, Brush, RectangleF, and other objects that you're used to working with, and just apply them to a new canvas.

That canvas is the C1PdfDocument object. You can draw directly to this object with the DrawString method for text, or add images with the DrawImage method. It also shares methods for drawings graphics with the regular .NET Graphics class. Not everything here maps quite so directly to existing .NET functionality, because ComponentOne does make allowance for some unique PDF functionality. For example, there are methods for adding bookmarks, links, and file attachments to PDF files. In theory, you can create any PDF document with this product, because there's a low-level .Write method that lets you emit PDF code directly into the output file. Most developers, though, probably won't want to deal with the PDF specification at that level of detail.

In addition to working directly with the C1PdfDocument object, you may choose to use any tool that can render content to a metafile or an RTF file, because ComponentOne PDF can use those two types of content directly to create PDF pages. The software supports multiple page sizes within a single PDF file, as well as setting PDF security properties. A set of samples in C# show the basics of using the component, and there are both standalone and integrated help files available. ComponentOne offers a variety of support options from newsgroup support through purchased incident support. The product can be purchased as a standalone component or as part of the more comprehensive ComponentOne Studio Enterprise suite of components.

FyTek's PDF Forms

Version 3.1 Windows EXE, DLL, .NET Version, $99.95
www.fytek.com/cgi-bin/getdemo.pl?prod=FRM
Licensing: Proprietary, per-user or per server

FyTek's PDF Forms product, despite the name, has nothing to do with PDF form fields. Rather, it provides a novel way to create PDF documents by merging an image file with a text file. You start with a JPG or TIF image and a text file laid out so that the information lines up with where you want it on the finished page in a fixed width font. Then you feed them both into FyTek's PDF Forms, and a PDF file comes out the other end. Think of it as an all-electronic analog to running an old paper form through a dot-matrix printer and you'll get the idea. The product includes three versions of the software: an executable controlled by command-line options, a standard Windows DLL, and a .NET library. The latter two work by letting you create a single object, setting lots of properties, and calling a single method to create the output.

Documentation in the form of a PDF file includes all the parameters and some sample code, and there is some additional sample code available for download along with a trial version of the program. Support is by e-mail, with 90 days free and extended support contracts available for purchase.

FyTek's PDF Meld

Version 5.2 Windows EXE, DLL, .NET Version, $29.95
www.fytek.com/cgi-bin/getdemo.pl?prod=MLD
Licensing: Proprietary, per-user or per server

PDF Meld is a batch-oriented application for manipulating PDF files. It lets you perform a variety of operations, including merging files, extracting pages, resizing and rotating page contents, adding annotations or bookmarks to a PDF file, adding page numbers, highlights, and graphics, adding or saving form data, and embedding or extracting files. PDF Meld comes as an executable driven by switches and command files, as a Win32 DLL, and as a .NET library. The latter is basically a single object with a whole mess of methods that match the switches for the command-line version.

Documentation in the form of a PDF file includes all the parameters and some sample code, and there is some additional sample code available for download along with a trial version of the program. Support is by e-mail, with 90 days free and extended support contracts available for purchase.

FyTek's PDF Report Writer

Version 3.2 .NET Version, $299.95
www.fytek.com/cgi-bin/getdemo.pl?prod=RWN
Licensing: Proprietary, per-user or per server

PDF Report Writer is not, despite the name, a databound reporting tool. Rather, it lets you create PDF files by supplying their definition in plain-text files using an HTML-like markup language. These markup files are read by the .NET library, which spits out the resulting PDF when you call its buildReport method. This method of building PDF files has the advantage that designing the file is not itself a coding activity; you need to learn a proprietary markup language, but PDF development under this system is more akin to Web design than writing .NET code.

The markup language supported by PDF Report Writer is quite comprehensive. It includes support for running headers and footers, nested tables, imported images with transparency, graphics including a sophisticated charting language, and PDF form fields. You can create annotations and bookmarks and include embedded files in the generated PDFs. Generated PDFs can be output to files or to Web pages on the fly.

Documentation in the form of a PDF file includes all the parameters and some sample code and report definitions. You can download a trial version to investigate further. Support is by e-mail, with 90 days free and extended support contracts available for purchase.

FyTek's Text2PDF

Version 3.1 .NET Version, $19.95
www.fytek.com/cgi-bin/getdemo.pl?prod=T2PN
Licensing: Proprietary, per-user or per server

As you can probably guess from the name, Text2PDF converts text files to PDF files through a simple object model - basically, read the text file into an object, set a few options, and call the BuildPDF method. Simple angle-bracket proprietary markup can be used to add a bit of PDF sugar to the results, including bookmarks, hyperlinks, images, and font settings. By and large, though, what you put into the plain text is what's going to come out the other end as the PDF stream.

Documentation in the form of a PDF file includes all the parameters, as well as some sample text files demonstrating the use of the various parameters that you can embed. There's also more sample code for downloading. You can download a trial version to investigate further. Support is by e-mail, with 90 days free and extended support contracts available for purchase.

eDocEngine ActiveX/.NET

Version 2.0, $299
www.gnostice.com/edocdownload_x.asp
Licensing: Proprietary, per developer with redistribution rights

Gnostice's eDocEngine is a general-purpose document-generation engine implemented as an ActiveX control (which also works on .NET Windows forms). There's also a version built as a .NET DLL. It includes PDF as one of its output formats, but it is also able to generate HTML, Excel, RTF, TIFF, SVG, and many other file formats. It uses a very flat object model to do this; you instantiate an appropriate engine object (gPDFEngineX in the case of PDF files) by dropping the control on your form, and then call methods of the control such as TextOut or DrawImageXY to produce the document. Helper classes handle some higher-level abstractions such as tables, annotations, and borders.

Help is supplied in the form of a single help file and a demo; the help file also includes a FAQ and sample code. You'll probably still have to spend some time reading through the long list of methods and properties to figure out how to make eDocEngine do what you want. Support is via e-mail.

PDFtoolkit ActiveX/.NET

Version 2.0, $299
www.gnostice.com/pdftoolkitdownload_x.asp
Licensing: Proprietary, per developer with redistribution rights

The Gnostice PDFtoolkit supports a range of editing operations on PDF files. It's supplied as an ActiveX control (which can also be hosted on .NET Windows Forms) and as a .NET library). The control lets you view or print PDF files without having Acrobat Reader installed. With this software, you can fill in or read form fields, add watermarks, rearrange pages, split or merge documents, add bookmarks, or read the document properties from a PDF file. You can also extract the text or images from pages, insert new text, activate URLs, and flatten form fields. Everything is done through a main gPDFDocumentX class, with some helper classes to handle things like watermarks and encryption. The altered documents can be saved to a file or a stream.

Help is supplied in the form of a single help file and a demo; there's also sample code in VB.NET and C#. Support is via e-mail.

iTextSharp

Version 0.04, Free
http://sourceforge.net/project/showfiles.php?group_id=72954
Licensing: LGPL

iTextSharp is a port to C# of the Java iText library for generating PDF files. On the one hand, this means that despite the low version number, iTextSharp encompasses a great deal of functionality. On the other, it also means that it tends to use Java idioms rather than .NET ones, so developers with a strictly .NET background may find it somewhat difficult to use at times. For example, rather than setting a font property, you specify fonts using a method of the FontFactory:

```
Paragraph p1 = new Paragraph(new Chunk("This
is my first paragraph.",
FontFactory.getFont(FontFactory.HELVETICA,
12)));
```

To use iTextSharp, you create a Document object, and then create a PdfWriter that listens to the Document and directs its output to a stream which could be a FileStream or some other type of stream entirely. You then use methods of the Document to add content, for example:

```
document.Add(new Paragraph("Some text"))
```

When you close the document, the writer takes everything you've dumped into the document and outputs it as a PDF file.

iTextSharp can handle quite a bit of advanced PDF formatting including embedded TrueType fonts, tables, links, outlines, graphics, and more. However, the documentation is limited to a set of online tutorials, and support is via the SourceForge discussion board for the project only. Although the discussion board is still active, the library itself has not been updated since August 2003, and it seems likely that it is no longer under active development.

iText.NET

Version 1.2.1-1, Free
http://sourceforge.net/project/showfiles.php?group_id=78685&package_id=79924
Licensing: MPL/LGPL

iText.NET is a port to J# of the Java iText library for generating PDF files. Note that the port is to J# rather than to C#, which means that you take on several additional dependencies if you use this library; you need to include the J# redistributables as well as the CLASSPATH libraries from the GNU Foundation. You'll want to look closely at the open source licensing and copyright requirements of the various code pieces involved. You should also be prepared for a Java-ish way of doing things at times, though the author has added some C# wrappers to the underlying iText functionality.

To use iText.NET, you create a Document object, and then create a PdfWriter that listens to the Document and directs its output to a stream (which could be a FileStream or some other type of stream entirely). You then use methods of the Document to add content, for example:

```
document.Add(new Paragraph("Some text"))
```

When you close the document, the writer takes everything you've dumped into the document and outputs it as a PDF file.

iText.NET can handle quite a bit of advanced PDF formatting including embedded TrueType fonts, tables, links, outlines, graphics, and more. For documentation, you're referred to the original iText tutorials. There's also a fairly extensive set of sample code translated into C# and J#. The port is under active development, and releases trail the Java version by only a couple of weeks. Support for the port appears to be minimal, with almost no discussion on the SourceForge site.

PDF4NET

Version 2.6, $499
www.o2sol.com/downloads.htm
Licensing: Proprietary, per-developer with redistribution rights

PDF4NET is a 100% managed code library with versions for the full .NET Framework and the .NET Compact Framework. It can function as a PDF generator, but it also includes full PDF editing and PDF forms functionality. For PDF generation it uses an approach similar to the implementation of GDI+ in the System.Drawing namespace. The PDFPage object exposes a Canvas on which you can draw text, images, and so on. The objects and methods in PDF4NET are broadly similar to those in System.Drawing but not identical. For example, PDF4NET supplies its own PDFPen and PDFBrush objects rather than reusing the existing Pen and Brush objects). After you've finished drawing the pages in your document, you can save the document to a file or directly to an ASP.NET output stream.

PDF4NET supports a good variety of PDF features, including text (regular and HTML-formatted), graphics, bookmarks, and annotations. You can make use of either 40- or 128-bit security. You won't find some of the higher-level abstract objects supported by other libraries such as tables and bar codes; if you need those objects, you'll need to build them up yourself.

On the PDF editing front, the software can add new pages to an existing document, or add new content to an existing page, but it can't alter existing content. You can overlay or underlay new content on existing pages as a fresh layer. You can also merge two PDF files into one, split a PDF file into individual page files, apply security to an existing file, or extract the full or partial content from one PDF file to another. PDF4NET also lets you programatically fill in a PDF form and save the results in a file, as well as flatten form fields to static text.

Along with the libraries, PDF4NET installs a good help file and plenty of samples written in C#. Support is provided via a Web site accessible to registered customers only. The base price includes 90 days of support, and you can purchase longer support contracts in one-year increments.

PDFTron PDF PageMaster

Version 1.1, Contact vendor for pricing
www.pdftron.com/downloads.html
Licensing: Proprietary

PDF PageMaster is primarily a library for splitting and merging PDF files, and rearranging pages into new PDF files. With it, you can merge PDF files, rearrange the pages in an existing file, or append pages to a new file. There are high-level methods to do things like quickly extract page ranges or all even pages from an existing document and save them to a new document, and the output can also be directed straight to the ASP.NET output stream. In addition, PageMaster supports setting document properties such as author, subject, title, and keywords, and setting document security and encryption.

PDF PageMaster installs as separate libraries for .NET 1.0 and 1.1, together with a Word document that explains the PageMaster API. There's also sample code for both desktop and ASP.NET use. In addition to the .NET version, PDFTron can also supply PageMaster as a GUI program or as a COM library. You can download the software for evaluation. Support is via e-mail for 30 days, with support contracts available for purchase.

PDFTron PDF2SVG

Version 1.1, Contact vendor for pricing
www.pdftron.com/downloads.html
Licensing: Proprietary

PDF2SVG is a .NET library with a single purpose. You instantiate the Pdf2SvgComponent, set properties controlling such things as whether you'd like embedded fonts and JavaScript, and call the Process method. The result is your PDF converted to SVG (Scaleable Vector Graphics), a standard format for high-fidelity graphics on the Web. Depending on the options you choose, PDF2SVG will wrap the whole thing up in an HTML e-book format so you can publish it straight to the Web for others to read (and you can even customize the HTML to fit the look and feel of your own site). Although it is a W3C standard, SVG is not currently widely supported by default in many browsers. Of the major browsers, only Opera currently offers native SVG support. However, this appears to be changing, with Firefox 1.1 promising SVG support. Internet Explorer can view SVG files with the aid of a browser plug-in from Adobe.

PDF2SVG comes with sample code in both C# and VB.NET. Documentation of the API is available online at the PDFTron Web site. In addition to the .NET version, PDFTron can also supply PDF2SVG as a GUI program or as a command-line program. You can download the software for evaluation. Support is via e-mail for 30 days, with support contracts available for purchase.

PDFNet SDK

Version 2.12, Starting at $99
www.pdftron.com/downloads.html
Licensing: Proprietary

The PDFNet SDK is an amazingly comprehensive library for working with PDF files at all levels, from building simple PDF viewers to creating new PDF files from scratch to editing existing files. It offers access to Adobe's low-level APIs through its own 100% managed code objects, as well as its own higher-level abstractions that allow powerful operations that would be incredibly difficult at the low level. Editing PDF files includes the capabilities found in most other products - merging, splitting, adding new content to pages, and moving pages around. But there's also an elegant API to edit existing text and graphics; it only takes a few lines of code, for example, to navigate down into the text on a page, read what's currently there, and make changes. You can extract individual elements from the page as well.

The PDFNet SDK also allows you to interact with form fields. You can create new fields, fill out existing fields, and flatten fields to static content. There are APIs for viewing, rasterizing, or printing pages from PDF files. You can manage encryption and document properties and extract or embed fonts. The API concentrates on complete support of the low-level standard, so if you want high-level notions like tables you'll need to build them up yourself. All versions of the PDF specification right up to the latest 1.7 are supported. PDFNET is delivered as a pair of .NET libraries (one each for the 1.0 and 1.1 frameworks) and a help file that's shared with the unmanaged C++ version of the library (so it does not follow the usual MSDN conventions). There is plenty of sample code included as well, in both VB.NET and C#.

PDFNet SDK licensing depends on your functionality requirement and type of deployment, not your development team. You can choose from among six different license types depending on which features you need, from simple read-only access to the low-level PDF API to the full high-level editing API. Depending on your deployment size and type, you can choose from CPU or OEM Redistribution Licensing. Custom-tailored licenses are also offered. PDFTron appears to want to only sell you the functionality you need, which is refreshing You can download the software for evaluation. Support is via email for 30 days, with support contracts available for purchase.

PDFXpress Professional

Version 1.0, $1499
www.pegasusimaging.com/cgi-bin/download2.cgi?PDFXR1

Licensing: Proprietary, per-developer and per-user

PDFXpress provides a .NET Windows Forms control, implemented in 100% managed code, that you can use to implement a PDF viewer in your own applications. To use it, you place the control on a form, where it appears in the component tray. At runtime, you add a PDF document to the control's Documents collection. You can then call the RenderPageToBitmap method to get a bitmap version of any page in the document. You can set the resolution of the rendered bitmap to make your own tradeoff between speed and accuracy. PDFXpress supports the entire PDF standard; it had no trouble rendering large and complex PDF files in our tests.

When you install PDFXpress, you get MSDN-style help integrated with Visual Studio .NET as well as several sample applications. Pegasus provides support via e-mail and newsgroups; you can purchase a priority support agreement to receive telephone support. The full product is available for download, with a license required for distribution.

Report.NET

Version 0.08.01, Free
http://sourceforge.net/project/showfiles.php?group_id=58374
Licensing: LGPL

Despite the name, Report.NET is not a general-purpose reporting product. Rather, it is an open-source C# object library for building PDF documents from scratch. You create a report object, add page objects, and then call methods of the page objectto add content to the page. A PdfFormatter object determines the output stream that will receive the finished document, which can be directed to a file or to an ASP.NET output stream. A helpful ReportTools class provides a static ViewPDF method for viewing PDF files in the Acrobat viewer.

The library supports text, lines, rectangles, and jpg images. You can format data as text flowed from page to page, or as tables, using layout managers defined for that purpose. Documentation with the software is limited to a few samples and XML comments in the C# source code; if you run the latter through a utility such as NDoc you'll get a start at a help file, but you'll certainly need to spend some time figuring out how it all fits together. Support is via the SourceForge discussion group, with most questions picking up an answer within a few weeks.

SharpPDF

Version 2.0, Free
http://sourceforge.net/project/showfiles.php?group_id=106579
Licensing: LGPL

SharpPDF is an open-source PDF generator written entirely in C# and licensed under the LGPL. For this Guide, I looked at Beta 2 of version 2.0, which was nearly ready to be declared the release version. To use SharpPDF, you first create a pdfDocument object and then add other objects to this object. These include both resources (such as fonts and images) and parts of the document (such as pages, text, paragraphs, and tables). Each element has properties that control its appearance. For example, when you create a paragraph you can control its font, X and Y coordinates, color, border, height and width, and so on.

One nice feature here is built-in support for flowing paragraphs and tables from page to page. This is implemented with the notion of "checked" elements. When you create a checked paragraph, for example, SharpPDF formats as much text as will fit into your specified bounds and then returns the overflow to you in an output string so you can flow it into your next layout element. SharpPDF also implements a number of "abstract objects" that are built up of simple elements. In addition to tables, these include page markers (such as "page n of m") and persistent page elements such as headers and footers. You can also define PDF controls, which are the analog of Windows Forms controls: compiled controls that you can reuse in other SharpPDF documents.

As with many open-source projects, SharpPDF is marred by sketchy documentation. The source code includes XML comments explaining the various classes and members, and the author has used NDoc to build these into MSDN-style help for the library. There's also a small set of code samples available on the SharpPDF Web site. Support is via the SourceForge discussion board for the project, which seems to have a reasonable number of participants; most questions get answered promptly and helpfully.

Essential PDF

Version 3.0.1.0, starting at $495
www.syncfusion.com/Downloads/evaluation.aspx
Licensing: Proprietary, per-developer or per-server

SyncFusion's Essential PDF is part of its Essential Suite of .NET libraries, although it can be purchased separately (with or without source code). It provides an object model for building PDF files that's somewhat loosely patterned on the Graphics model used by the .NET drawing classes, though with some extensions. For example, you can create a table by importing an ADO.NET DataTable, which is convenient for publishing data from a database to a PDF file (though it obviously won't get you all the flexibility of a full-blown banded report writer). Annotations, bookmarks, hyperlinks, images, graphics, and PDF security are among the other PDF features supported by this library.

By building the library to mimic the System.Drawing classes, Syncfusion has made it easy to perform a number of operations that are tough with some other alternatives. You get a good variety of graphics primitives that are easy to place on the page, scale, and rotate. Text, too, can be rotated, as well as constrained to a particular rectangular region. Images support both transparency and masking.

Essential PDF is still clearly a work in progress; the Getting Started guide even labels it a beta product and contains a fair number of typos. Despite this, there is solid MSDN-style help for the entire class library integrated into Visual Studio's help, as well as a good chunk of sample code to learn from. Syncfusion also supplies peer forums and an incident response system for registered users to receive support. You can download a trial copy as part of the company's Essential Suite product.

PDFKit.NET

Version 1.0, starting at $899
www.tallcomponents.com/Default.aspx?id=pdfkit-download
Licensing: Proprietary, licensed per server or application

PDFKit.NET provides an object-oriented API for reading and manipulating PDF files and forms on the fly. You can read in documents from a stream, and write the results out to anything that will take a BinaryWriter which means it's easy to output the results to a file or via ASP.NET to a browser. Within the PDFKit.NET object model, you can merge and split documents, append pages, shrink pages to build pages with multiple thumbnails, stamp new content (text, images, or graphics) on to existing pages as an overlay or underlay, and set document security properties. You can also fill out and flatten PDF form fields. There's MSDN-style help for everything, plus tutorial material and an ASP.NET sample application.

PDFKit.NET is licensed according to your deployment model, not your development team. Rather than buying licenses for each developer, the license purchased depends on whether the software will be used on a server, incorporated in a client application, or resold as part of a client application to your own customers. You'll need to contact TallComponents for detailed pricing information depending on your own particular deployment scenario. Support is via e-mail; you can also purchase priority support on an annual per-developer basis.

Fully-functional evaluation copies are available from the company's Web site.

PDFRasterizer.NET

Version 1.0, starting at $899
www.tallcomponents.com
Licensing: Proprietary, licensed per server or application

PDFRasterizer.NET allows you to convert PDF files to images using an extremely simple object model. You load the PDF into the Document object, navigate through the Pages collection to find the Page object that you want, and then call the Page.Draw method. This method takes a Graphics object as its target; this makes it compatible with an image, a Windows Form surface, or a printer right out of the box, thanks to the general-purpose nature of the .NET Graphics class. The library comes with sample code in both C# and VB .NET showing how to develop PDF viewers and printers as well as how to convert a PDF to a series of images. MSDN-style help rounds out the offering.

PDFRasterizer.NET is licensed according to your deployment model, not your development team. Rather than buying licenses for each developer, the license purchased depends on whether the software will be used on a server, incorporated in a client application, or resold as part of a client application to your own customers. You'll need to contact TallComponents for detailed pricing information depending on your own particular deployment scenario. Support is via e-mail; you can also purchase priority support on an annual per-developer basis. Fully-functional evaluation copies are available from the company's Web site.

TallPDF.NET

Version 2.0, starting at $499
www.tallcomponents.com/default.aspx?id=tallpdf-download
Licensing: Proprietary, licensed per server or application

TallPDF.NET is a managed code library that allows you to create PDF documents through an extremely rich object model. In addition, it features transformation of XML documents into PDF and a unique event-driven PDF generation mode that's similar to the way that the .NET printing model works. This three-pronged approach gives you a good deal of flexibility when using TallPDF.NET to generate PDF documents from your application. PDF documents can be easily written to a file or other Stream object or sent directly to the ASP.NET Response object.

The object model in TallPDF is very fine-grained. For example, a document is broken up into sections, which each contain a collection of paragraphs, as well as such objects as OddHeader, OddFooter, EvenHeader, EvenFooter, and so on. Images and Tables are specializations of Paragraphs.

For the most part, you create a document by creating objects, setting their properties, and then adding them to the appropriate collections. All of this is very well documented in MSDN-style help. There are some nice extra touches here as well, such as support for automatically building tables of contents and lists of figures, and fields that let you resolve page numbers, section numbers, and so on dymamically at document build-time. The software also supports PDF encrypton if you buy a Professional license.

TallPDF.NET is also designed to create PDF documents directly from XML documents that are structured according to the TallPDF Document Object Model. This is as simple as creating an instance of the Document object, invoking its Read method on an appropriate XML document, and then invoking the Write method to write it back out as PDF. You can also read any element out of an XML document and use it to build part of a PDF, mixing this with class-based programming as your needs dictate. Of course, you can also use XSL transforms on the way in to convert your own custom XML to the format that TalPDF.NET is expecting.
The event-driven mode allows you to build a PDF document while TallPDF is actually generating it. This keeps memory consumption down by only requiring the resources for a single page at a time, which is useful in a heavily loaded environment. Similar to the way that .NET printing works, TallPDF fires events such as StartPage and PrintParagraph, and it's up to your code to feed in content when these events are fired. When using events, the layout engine is forced into "forward-only" mode which keeps you from using features like "Page X of Y." This limitation is to be expected from event mode but it can be worth it when you need processing speed and/or lower resource usage.

TallPDF.NET is licensed according to your deployment model, not your development team. Rather than buying licenses for each developer, the license purchased depends on whether the software will be used on a server, incorporated in a client application, deployed as part of an ASP.NET application on a Web farm, or sold as a Web application to your own customers. You'll need to contact TallComponents for detailed pricing information depending on your own particular deployment scenario. Support is via e-mail; you can also purchase priority support on an annual per-developer basis. Fully-functional evaluation copies are available from the company's Web site.

Visual Programming Ibex PDF Creator

Version 3.0, starting at $795
www.xmlpdf.com/ibex-downloads-net.html
Licensing: Proprietary, licensed per developer, redistribution with limitations

Ibex PDF Creator is an XSL-FO Formatting Engine that takes XML in the XSL-FO format defined by the W3C, chews on it, and outputs PDF files. Alternatively, you can take raw XML and an XSLT stylesheet with XSL-FO directives and feed them both in to Ibex PDF Creator to get both the transformation and formatting performed by this engine; the result is still PDF. Often this is the technically preferred approach, as some things (such as numbering headings) are difficult or impossible to do directly in the final XSL-FO format.

Ibex implements the full XSL-FO specification as defined in the W3C XSL 1.0 recommendation, along with some extensions developed by Visual Programming. These extensions allow you to handle PDF security, bookmarks, and document properties, among other things. You get a command-line interface as well as a simple API you can call from your own applications to perform the transform and formatting steps.

For the most part, the API only requires a single call to doc.Generate to do the heavy work. The vendor has tested this software in quite demanding situations: it can handle documents up to the 2GB size limit for PDF files and embed SVG graphics in their native vector format for lossless zooming. The company has published performance figures on their Web site for documents up to 80,000 pages long which they claim take about 4 hours to generate.

Ibex PDF Creator comes with an extensive manual that includes a good tutorial on XSL-FO and some sample code. You can view additional samples on the company's Web site. Licensing is per-developer with free support; maintenance contracts are available for priority support.

General Criteria

What things should you consider when purchasing any type of third party development tool or component? Or when deciding between purchasing such a component and building it in-house? Here are some factors you might like to think about:

• Can you download an evaluation or demo copy prior to purchase?

• What are the limitations to the evaluation or demo copy?

• What are the company's tech support policies? How is tech support offered (phone, e-mail, newsgroup, discussion board)? During what hours? What's the average response time? Is an answer guaranteed? What does support cost? How long do you get free support? Can you purchase a support contract and what will it cost if you can?

• What are the company's return policies?

• What form is documentation provided in? Text file? PDF file? Help file? HTML Help file? Integrated to Visual Studio help? Printed manual?

• What architecture is the product? Pure managed code, pre-wrapped ActiveX, wrappable ActiveX, non-managed code designed to be called directly from managed code?

• What other products does this vendor offer that you might need? All other things being equal, sticking with products from one vendor can lower your support costs and ease the learning curve associated with new components.

• Can you develop the functionality you need in a cost effective time-frame? Do you want to support the functionality after you first development it?

• If you build yourself, how many other developers will be using it or will it just be you? If more than just yourself, do you have the time to write good documentation?

• Does the product support relevent standards where applicable, or does it use propriety implementations?

• If open source, is the project active? If not, you might end up supporting and/or enhancing it yourself.

• If commerical, how large is the vendor? How long have they been in business? How focused is their product line? How long have they supported this product? Have they ever dropped support of other products? Do they offer source code?

Note: Be careful with very large vendors that are not focused in the area of your interest. Large vendors have a bad habit of becoming interested in developer tools yet quickly drop support when they realize how hard it is to make money selling components and tools to developers. Exceptions are when the developer tools are their core competency or support their strategic direction. Conversely, don't discount small vendors if they have been in business for a while and have shown a proven ability to focus and provide quality products.

Products Not Covered in this Guide

These are the products appropriate for this Guide but for one reason or another we did not cover. In some cases, we simply didn't locate the products in time to include them in this edition of the Guide, and hope to add them to the next revision. In others, they declined to participate and refused to send us a license for evaluation. We list these products and their URLs here so you can research them on your own if you like.

ActivePDF - Vendor declined to participate citing negative past experience with similar projects produced by other organizations, and they voiced concerns our writers might be biased.

Publisher's Note: *As far as we know, there have not been similar projects covering PDF components (i.e. selections guides vs. product reviews), and since our goal is to achieve total respect from the .NET development community we'd be shooting ourselves in the foot if we allowed bias in this guide.*

Website: http://www.activepdf.com/

Altsoft Xml2PDF - We did not learn of this product until after the first draft was complete so we did not have a chance to contact Altsoft about including Xml2PDF.

Publisher's Note: *We hope to include Altsoft Xml2PDF in a future revision of this guide.*

Website: http://www.alt-soft.com/

Chive Apoc XSL-FO - Chive claims they "are not actively developing or marketing these products any longer."

Publisher's Note: *Chive however still solicites purchase of this product at http://www.chive.com/HowToPurchase.aspx*

Website: http://www.chive.com/

Chive PDF Toolkit - Chive claims they "are not actively developing or marketing these products any longer."

Publisher's Note: *Chive however still solicites purchase of this product at http://www.chive.com/HowToPurchase.aspx*

Website: http://www.chive.com/

Divido PDF - We only learned of Divido PDF in the final days of editing this Guide.

Publisher's Note: *We hope to include Divido PDF in a future revision of this guide.*

Website: http://www.dividosoft.com/

ECRION XF Rendering Server 2005 - We only learned of ECRION XF Rendering Server 2005 in the final days of editing this Guide.

Publisher's Note: We hope to include ECRION XF Rendering Server 2005 in a future revision of this guide.

Website: http://www.ecrion.com/

ED Components PDF Component - We only learned of ED Components PDF Component in the final days of editing this Guide.

Publisher's Note: *We hope to include ED Components PDF Component in a future revision of this guide.*

Website: http://www.edcomponents.com/

RareFind PDF N MORE for .NET - We only learned of RareFind PDF N MORE for .NET in the final days of editing this Guide.

Publisher's Note: *We hope to include RareFind PDF N MORE for .NET in a future revision of this guide.*

Website: http://www.rarefind.com/

Siberix PDF Library - Siberix PDF was released during the final days of editing this Guide.

Publisher's Note: *We hope to include Siberix PDF Library in a future revision of this guide.*

Website: http://www.siberix.com/

webSupergoo ABCpdf.NET - The principle at webSupergoo had a medical emergency within his family and was not able to participate.

Publisher's Note: *Our best wishes go out to the family member and our hope is they are able to make a full recovery very soon.*

Website: http://www.websupergoo.com/

Visual Programming XMLPDF - Visual Programming stated XMLPDF was a legacy product compared with their IBEX PDF Creator product, covered in this guide, because XMLPDF supports a proprietary XML schema. Although Visual Programming recommends IBEX PDF Creator over XMLPDF they still offer XMLPDF available for sale.

Publisher's Note: *We would have liked to cover XMLPDF even though it was legacy but we respected Visual Programming's wishes, especially since they did not make a full version of XMLPDF available for evaluation.*

Website: http://www.xmlpdf.com/

About the How-To-Select Guides

The How-To-Select™ Guides Process

We begin the process of creating a Guide begins by identifying a product category we believe will be of interest to many .NET developers. Next, we identify every product we can find that fits into the category. We then invite commercial products to participate and require them to supply our writer(s) with a fully-licensed copy of their product(s). In the future we also plan to require a small research fee to help cover paying our writers, not to exceed US$500, but for this first guide and probably the next several we waived that fee. In addition when applicable open source and public domain product exists, we include them in our Guides and waive the participation fees for these products.

Editorial Policies regarding Vendors

At the beginning of the process, vendors are invited to submit their opinion of what a developer would need to know in order to select a product in the category. Our writer(s) then take that information and do their own research to produce an intial draft of a Guide. We then give vendors an opportunity to review this initial draft and to submit factual corrections to the sections regarding their own products before publication. The final draft of the Guide is then prepared by our editorial team and submitted to the printer. We do not allow vendors to provide input to final version of the Guide before it is published.

In additional, our policy is that once a vendor agrees to participate in a Guide, our decision on how to cover their product and what to say about it is final.

Funding the How-To-Select™ Guides

For the first few guides we made available only a single exclusive sponsorship to vendors in the Guides. These vendors supply their own promotional material for the sponsorship section as advertising content. On an ongoing basis we will solicit advertising in the Guides and allow us to to fund the level of research needed to make our Guides able to meet the needs of the most demanding developer.

Feature Charts

The feature charts in this Guide represent our best effort to pick out some of the most important features for this class of software and to indicate which products support which features. It's impossible to include every feature of every product in these charts, and it's impossible to capture the subtle differences between products with such crude delineations.

With that in mind, you should use the feature charts as general guides to help you find products that might meet your needs and not as the final word on a product's capabilities.

Editorial Policy regarding Published Prices

Many of the products that we cover have quite complex pricing structures. For example, there may be separate prices for developer, redistribution, workgroup, and server licenses, for individual and quantity purchases, for named and floating licenses, and for perpetual or annual licenses. There may also be separate support or upgrade fees. In addition, many vendors offer discounts when you purchase their products through a reseller, or make promotional pricing available under certain circumstances.

Our policy is simple: we list the basic undiscounted list price for the least expensive fully-functional version of each product that we include in a Guide. You should always check directly with the vendor, or with your reseller of choice, for the most complete and current pricing. All prices are in U.S. dollars unless otherwise indicated.

Reader Feedback

While we strive to make each How-to-Select™ Guide as complete and correct as possible, we recognize that nobody's perfect. Products change, people invent new techniques for solving old problems, and sometimes we just make mistakes. That's why we revise each Guide on a regular basis. And that's also why we invite your feedback to help make the next edition of this Guide even better than this one.

How-To-Select™ Guide Reader Forums

One way to get your feedback directly to us is to participate in our Web-based reader forums. You can visit the overall forum for the How-To-Select™ Guides at http://forums.howtoselectguides.com/ or the one for this PDF Guide specifically at http://forums.howtoselectguides.com/dotnet/pdf/. Any content related to the Guide is welcome, but we're especially interested in the following areas:

• What questions do you have that aren't answered in the Guide? We'll do our best to answer them!
• What new scenarios are you finding in your own work that we ought to include in the next edition?
• What other products should we consider including in the Guide?
• What areas of the Guide need correction or clarification?

In addition, we'd like you to upload your own "How-To-Select Case Studies." If you evaluate two or more of the products in this Guide, please take a few minutes to write up your scenario, what you learned, and why you chose the product that you did. Your fellow developers will thank you!

E-mail Feedback

If you prefer, you can also send your feedback directly via e-mail. If you've got feedback on any of our content, or just want to get in touch to tell us what you thought of our coverage, we'd love to hear from you. If you're willing to have your feedback reposted to the online forum, you can contact our writers and editors at feedback@howtoselectguides.com. If you prefer, you can use feedback-private@howtoselectguides.com to send private feedback that we won't repost to our forum. We regret that we cannot promise a personal reply to every comment, but we will carefully consider all feedback in preparing the next edition.

Copyright and Trademarks

TALL™ components

Finally, a .NET PDF component
that's easier to work with than most people

Our 100% **.NET PDF creation component** offers state-of-the-art simplicity. A pull-model calls back to your code – whenever it needs content – to complete a page. Allowing you to dynamically create PDF documents with the highest speed, the lowest memory consumption and **extreme scalability.**

We also provide tried and trusted 100% .NET components to **split, stitch, stamp, query** and **form-fill** encrypted PDF documents. And they're so simple anyone could use them, well almost.

Visit our website for live demos, online documentation and to download fully functional evaluation copies.

www.tallcomponents.com

THE .NET PLATFORM DESERVES MORE THAN WRAPPERS :-)

FEATURING: DYNAMIC PDF / NO THIRD-PARTY DEPENDENCY / 100% MANAGED / FULLY UNICODE / RIGHT-TO-LEFT TEXT / PASSWORD PROTECTION GENERATE FROM XML OR PROGRAMMATICALLY / GDI+ SUPPORT / EMBED ALL RASTER IMAGES / EMBED TRUETYPE FONTS / HYPERLINKS / CHARTS AND TABLES HEADERS AND FOOTERS / FORM-FILL / STAMP / SPLIT / APPEND / STITCH / ENCRYPT