



Adobe® Acrobat®



Forms System Implementation Notes

© 1998 Adobe Systems Incorporated. All rights reserved.

The information in this document is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Adobe Systems Incorporated. Adobe Systems Incorporated assumes no responsibility or liability for any errors or inaccuracies that may appear in this document. The software described in this document is furnished under license and may only be used or copied in accordance with the terms of such license.

Adobe, the Adobe logo, Acrobat, Acrobat Capture, Acrobat Exchange, and Distiller are trademarks of Adobe Systems Incorporated. Microsoft and Windows are registered trademarks of Microsoft in the U.S. and other countries. Netscape and Netscape Communicator are trademarks of Netscape Communications Corporation. All other products or name brands are trademarks of their respective holders.

Version History		
08 April 1997		Original version
30 April 1998		Updated

Forms System Implementation Notes

Introduction

Acrobat Forms is a group of extensions added to PDF in version 1.2, used by the Acrobat 3.0 products. These extensions allow PDF files to contain fields and buttons. These can be considered to be simply a new layer on top of a PDF file. The underlying PDF file may be created by any PDF producer such as the PDF Writer, the Acrobat Distiller application, or the Acrobat Capture application. The fields are subsequently added manually using Acrobat Exchange 3.0.

In addition to the forms data stored in PDF files, there is a need for a file format to use for transmitting forms data between a server and a client. The Acrobat products support both the HTML form format (i.e. MIME type `application/x-www-form-urlencoded`) and an Acrobat-specific Forms Data Format: FDF (MIME type `application/vnd.fdf`).

This document provides an overview of the various architectures for Acrobat Forms applications. It also compares HTML and FDF as the file format for Acrobat Forms applications.

Pre-requisites

Readers are assumed to know the following:

- The process of creating form fields and buttons, and modifying their properties. The Acrobat Exchange 3.0 on-line documentation covers these. In particular, the Forms Online Guide, accessible via the menu item Help > Forms.
- The process flow for HTML forms, on both the client and server. Technical bookstores generally stock a number of excellent books that cover this topic.
- To take full advantage of the capabilities of Acrobat Forms, a knowledge of FDF is required. FDF is described in an appendix of the Portable Document Format Reference Manual, which is available at <http://www.adobe.com/prodindex/acrobat/pubs.html>.

Other sources of information

The latest version of Acrobat Forms can be found at <http://www.adobe.com/prodindex/acrobat/acrforms.html>

There is an Acrobat Forms Resources page at <http://www.adobe.com/prodindex/acrobat/acrforms.html#formsres>

The FDF toolkit page is located at <http://beta1.adobe.com/ada/acrosdk/forms.html>

The basics

After a user has filled in an Acrobat form, they must click on a button whose action is a submit form action in order to submit the data to a server. The

format of the submitted data may be either HTML-compatible (urlencoded) or FDF. The selection of which format to use is made at the time the form is created, in the same dialog box in which the form's creator enters the URL to which the data is submitted.

If HTML is selected, the format is identical to and compatible with an existing HTML form. Existing CGI scripts for HTML forms may be used to parse data in this format.

If FDF is selected, the data format is, not surprisingly, FDF. There is a server library to help parse and generate FDF files. See ["Other sources of information" on page 3](#).

The URL to submit to is not restricted to the http scheme. It can also be the mailto scheme, e.g. mailto:someuser@somecompany.com

Note: To see the format of the FDF data that is being sent to the server, create a form and enter data into one or more of its fields. Instead of submitting this to the server, simply select the "Export Form Data..." menu item from the File menu of Acrobat Exchange.

Choosing the output format

This section describes how to choose between HTML and FDF formats when implementing an Acrobat forms application.

HTML support allows Acrobat forms to be used as a direct replacement for HTML forms. Choose HTML if you need compatibility with existing systems.

FDF supports various capabilities not possible with HTML. Choose FDF if you wish to take advantage of its additional capabilities, which include:

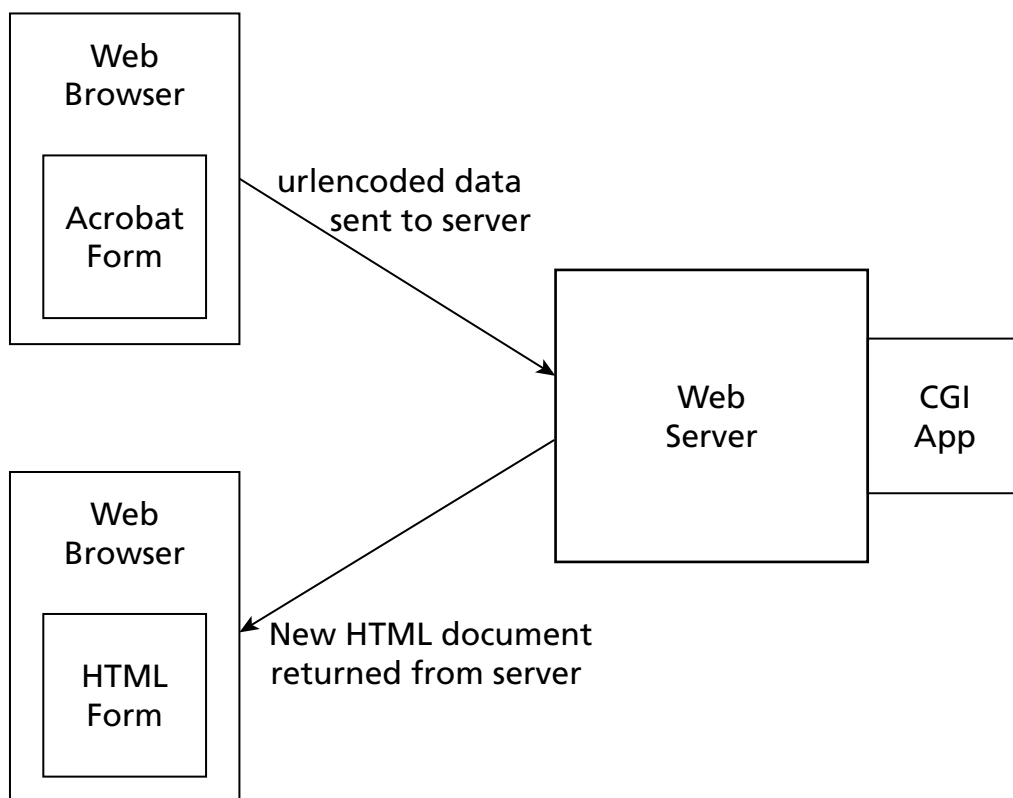
- When sending data back from the server, it is not necessary to re-send the form itself; the data can populate the same form that originated the data.
- Appearances (the visual look of a button). This allows you to change the way buttons appear. Additionally, you can take advantage of this capability to send graphical information in either direction between the client and the server.
- The form can be altered via an FDF sent back from the server: the various actions attached to buttons can be reprogrammed (including new Javascripts, change the URL for a submit form action, etc.); field properties can be changed, such as hidden, read-only, required, don't print; listboxes and comboboxes can be populated with different choices; etc.
- FDF can be used to dynamically synthesize PDF documents composed of a variable number of pages, from templates found in PDF documents specified by the FDF, and to populate any fields in the "spawned" pages with data. A template is simply a page with a name attached and may be visible in its source PDF document or hidden.

Replacing an HTML form with Acrobat form

[Figure 1](#) shows this simplest case, which permits existing CGI applications to be used without modification. To use Acrobat forms in such a system, you simply create:

1. An Acrobat form with field names that match those in the existing HTML form.
2. A button on the form whose action is a submit form action. The URL to submit to may be relative (to the URL of the form that you are submitting from).

Figure 1 Using HTML with Acrobat forms



Acrobat form with results returned in the same form

[Figure 2](#) shows a system in which the form data is returned into the same form as that from which it was submitted. This is a very commonly desired situation. In such a system, the data sent to the server may be either FDF or urlencoded format, while the data returned from the server *must* be in FDF format and have a MIME type of `application/vnd.fdf`.

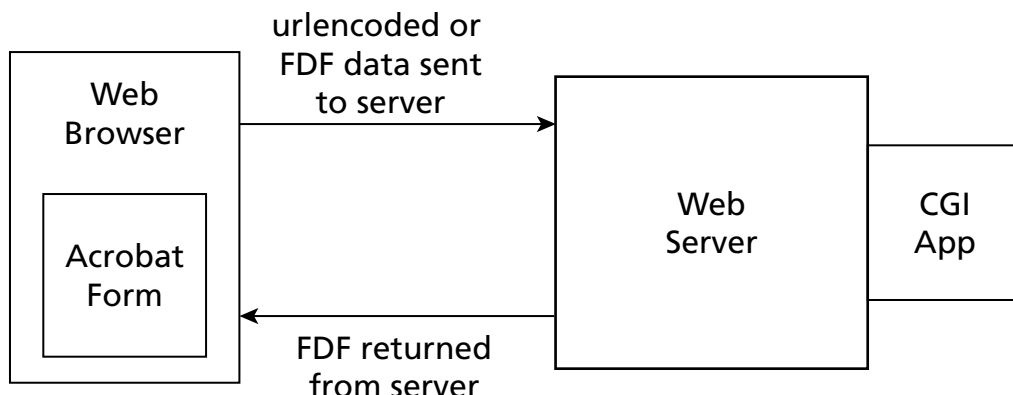
Note: When the server returns data in FDF format, the URL to submit to *must* end in `#FDF`, e.g. `http://yourserver.com/cgi-bin/yourscript#FDF`

Existing CGI applications must be modified to return FDF instead of HTML documents. [Example 1](#) shows a simple FDF-generating Active Server Page (ASP) that works with the Microsoft Internet Information Server 3.0. For anything more complicated than this example, the use of the FDF toolkit is highly recommended. See ["Other sources of information" on page 3](#).

Note: If you return a static FDF file stored on the server, as opposed to one dynamically generated by a server script as

in the example below, then you may have to define application/vnd.fdf as a new MIME type on your server.

Figure 2 Returning FDF into the same form



Example 1 Simple ASP for generating FDF

```
<%@ LANGUAGE = VBScript%>
<% Response.ContentType = "application/vnd.fdf" %>
```

```
Rem *****
Rem * Parse the inbound data *
Rem *****
```

```
Rem *****
Rem * Build FDF Stream *
Rem *****
```

```
%FDF-1.2
1 0 obj
<<
/FDF << /Fields [
<< /T (status)/V (Hello, World!) >>
] >>
>>
endobj
trailer
<<
/Root 1 0 R
>>
%%EOF
```

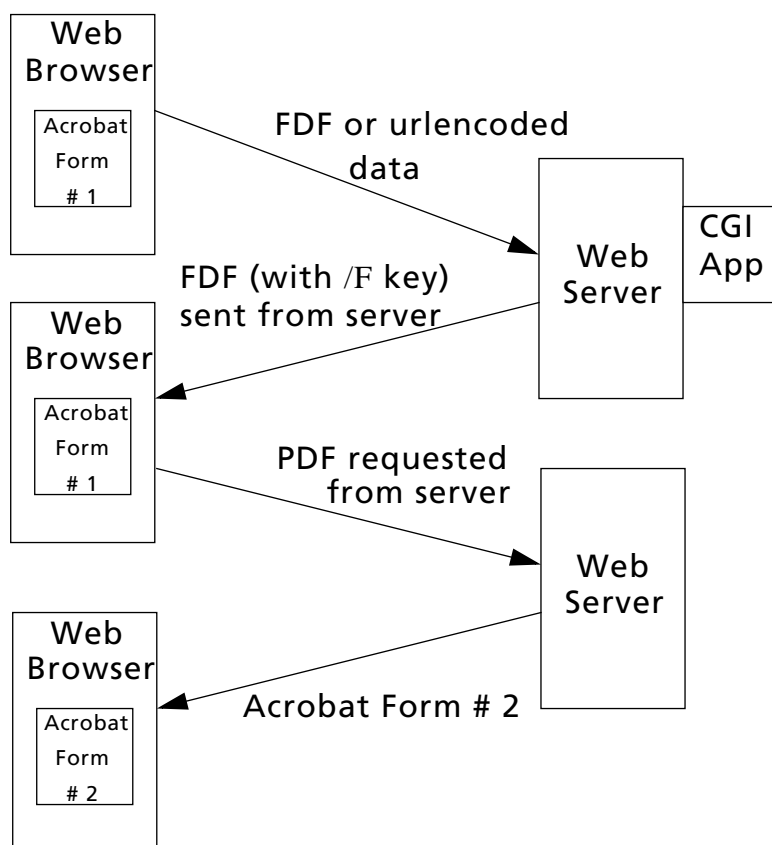
Acrobat form with results returned in a new form

In some cases, you may wish to return data into a different form, not the form from which it was submitted. [Figure 3](#) shows such a case. The modified CGI application needed to implement such an application is almost identical to that needed for the system described in the previous section. The only difference is that you must include an /F key in the FDF file when you want to populate a different form. The value of the /F key specifies the URL for the PDF file to populate with the forms data. This URL may be relative (to the URL of the form that you are submitting from). The specified file is retrieved (from the server) and populated with the forms data.

Note: If the returned FDF is for the same form that you submitted from, it should not include the /F key.

The data sent to the server may be either FDF or urlencoded format. The data returned from the server *must* be in FDF format and have a MIME type of application/vnd.fdf.

Figure 3 Returning FDF into a different form



HTML form with results returned in an Acrobat form

An additional possibility is starting from an HTML form, submitting to the server, and having the server return an FDF whose /F key gives as value the absolute URL of an Acrobat form. The specified form is retrieved (from the server) and populated with the forms data. [Figure 4](#) shows such a system.

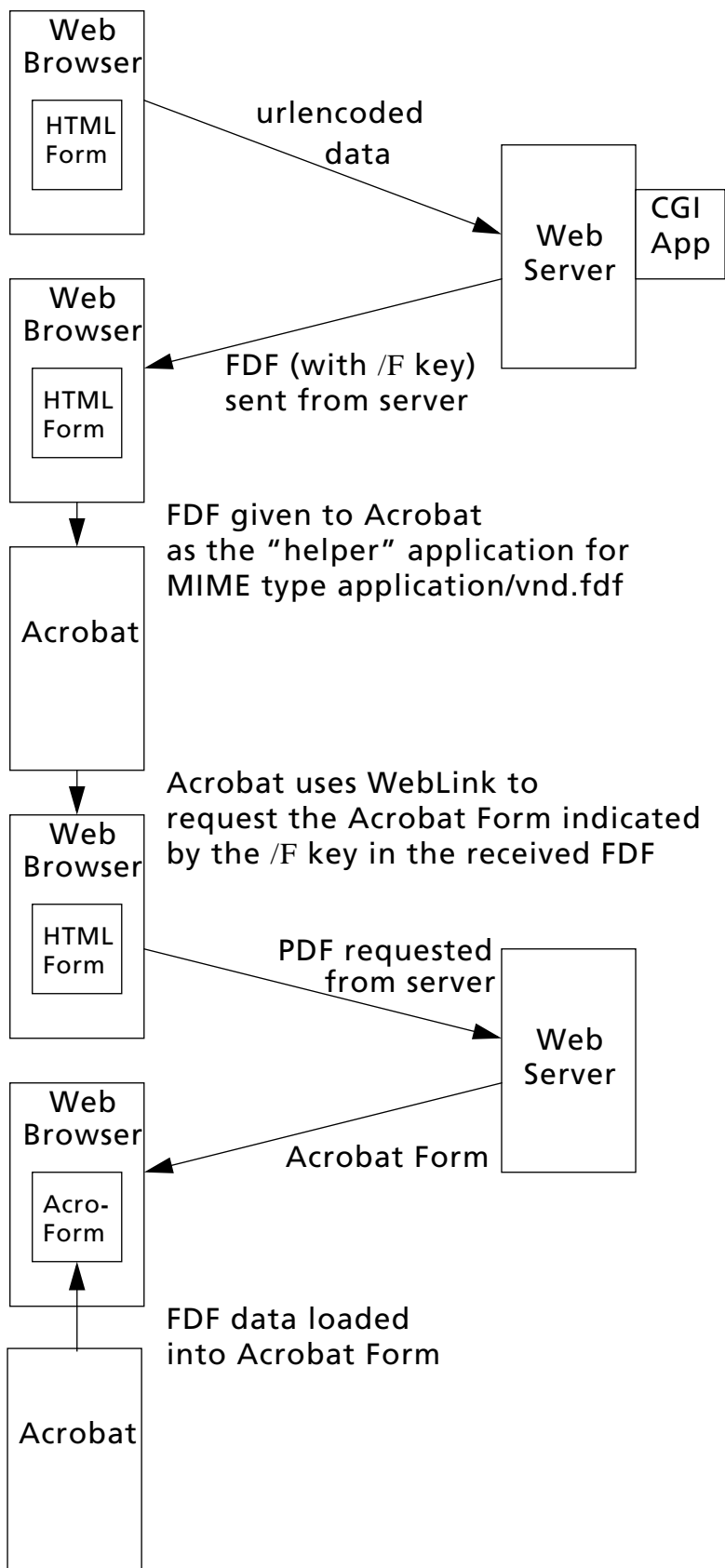
Note: For this to work, it is necessary that you select Acrobat as the application that handles the MIME type "application/vnd.fdf". For example, if you are using Netscape (e.g. Communicator 4.0x), then you do this under Preferences > Applications. Choose Acrobat as the "helper" application. If you are using Internet Explorer (on the Microsoft Windows platform), open Windows Explorer, choose the menu item View > Options > File Types, and make sure there is an entry for "Adobe Acrobat Forms Document" which has the "Default Extension for Content Type:" set to "fdf" and the "Content Type (MIME):" set to "application/vnd.fdf". Additionally, the checkbox "Confirm open after download" should be unchecked. Finally, under "Actions:" there should be an entry for "open", and in its properties, "Application used to perform action" should be set to C:\Acrobat3\Exchange\AcroEx32.exe "%1" (or

wherever Acrobat resides), and the checkbox "Use DDE" should be unchecked. It also is imperative that from within Acrobat you go to the File > Preferences > Weblink menu item and choose your browser.

Note: For the case being discussed in this section the URL to submit to does not need to end in #FDF, unlike the cases described before in ["Acrobat form with results returned in the same form" on page 5](#) and ["Acrobat form with results returned in a new form" on page 6](#)

Note: Under Internet Explorer the PDF will open in a new browser window from the one displaying the HTML that the submission was triggered from. Under Communicator 4.0x, it opens in the same window, and even in the same frame (if frames are being used).

Figure 4 Start from HTML, end in an Acrobat form



Templates

A recent addition to Acrobat Forms is the ability to dynamically create PDF documents composed of a variable number of pages, from templates found in PDF documents specified by the FDF, and to populate any fields in the "spawned" pages with data carried by that FDF. [Figure 5](#) shows such a system.

Figure 5 Dynamic creation of a PDF from templates

