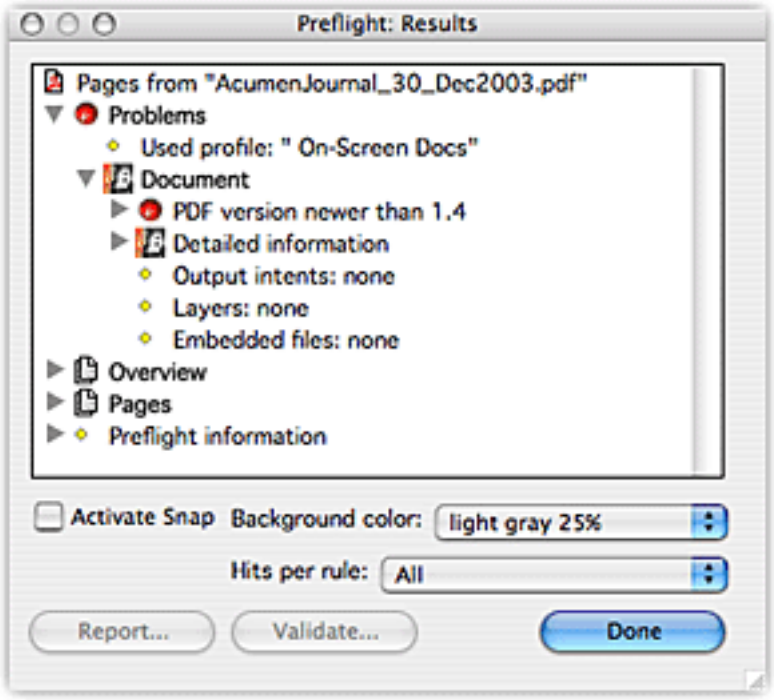
Table of Contents

Acrobat 6 Preflighting, Part 1



Among my favorite new Acrobat 6 features is the Preflight mechanism. Similar to features in products such as *PitStop,* Acrobat 6 Preflight scans the current PDF file, accumulating information and looking for user-defined problems.

The mechanism is very powerful and well worth learning. There is, however, a great deal to discuss, so we shall take two months to examine it all.

This month, we shall see how to conduct a preflight scan. We shall test a an issue of the Acumen Journal using a preflight profile named "On-screen Docs."

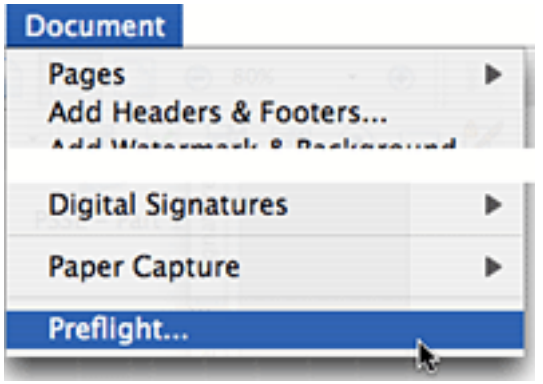
Next month, we shall see how to make our own preflight profiles. This is very much a necessity, since it allows you to look precisely for the specific problems common to your documents.

Next Page ->

Acumen Journal: Acrobat User 2

Using Preflight

You access the Acrobat 6 Preflight feature by selecting *Preflight* from the *Document* menu.



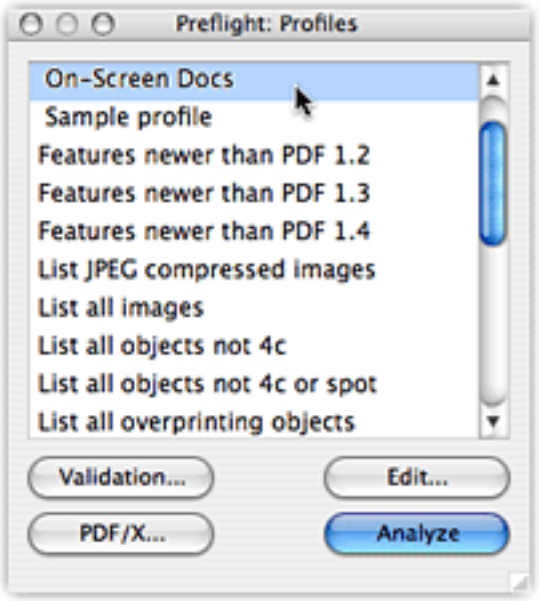
Preflight Profiles

When you do so, Acrobat presents you with the *Preflight Profiles* dialog box (below right). A preflight profile specifies the conditions that you consider to be problems in a file.

Thus a profile named "On-Screen Docs" might look for unembedded fonts, CMYK colors, and images with inappropriately high resolutions.

Although Acrobat ships with a set of default profiles, you will want to create your own so that your preflight scans can look for problems that are important to you. For example, the *On-Screen Docs* profile selected at right is what I use to test my *Acumen Journal* files.

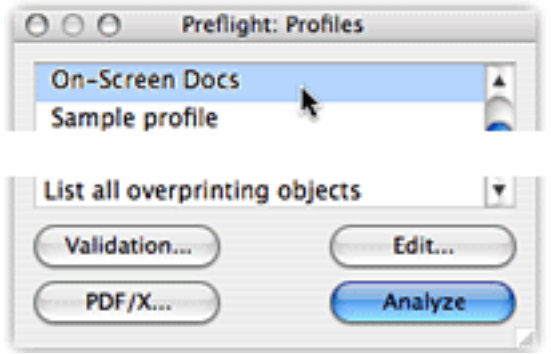
Again, we shall see how to create our own preflight profiles next month.



Next Page ->

Scanning a File

Having selected *Document>Preflight,* you run a preflight session on the current PDF document by selecting a preflight profile and then clicking the *Analyze* button.



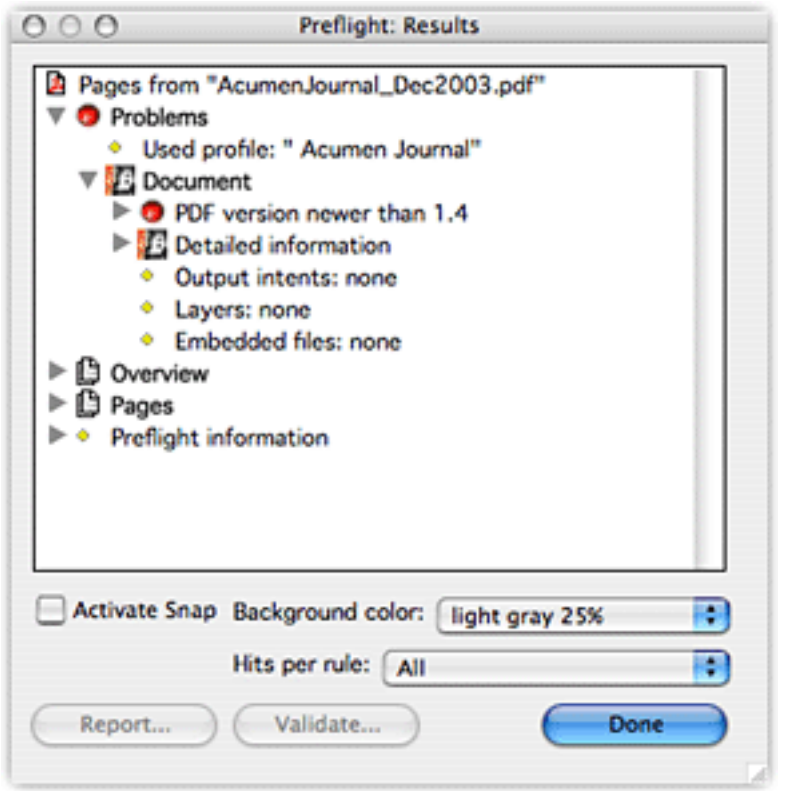
Acrobat scans through your file and then displays the *Preflight Results* dialog box (below right), that presents all the information gathered during the scan.

There are four types of information listed in the dialog box:

- Problems with the file
- An overview of information that applies to the entire file
- A page-by-page view of font usage, images, and other components of the file
- Information describing the preflight session, itself.

Let's look at these in detail.

Next Page ->>



Profiles for Downloading

The file *preflight.zip,* on the Acumen Training Resources page, contains two preflight profiles you can download and import into your copy of Acrobat:

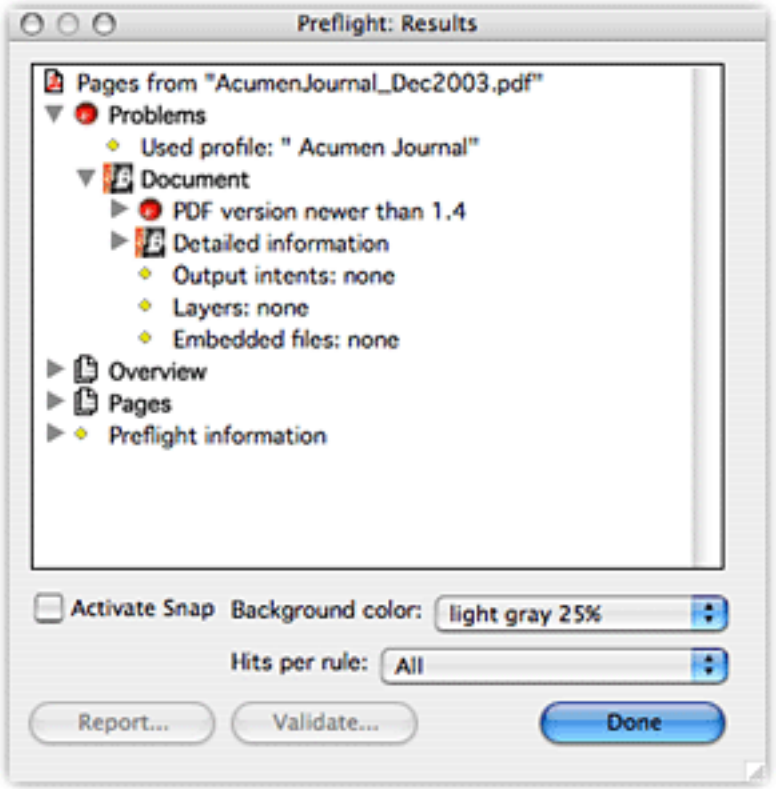
- *On-line Docs,* for use with Acrobat files that will be viewed primarily on-line. (This is the profile we use in this article.)
- *600-dpi,* appropriate to Acrobat files that will be printed on a typical high-speed laser printer.

Also included is a file containing directions for how to import a profile.

The Analysis Report

Let us look at the result of scanning an issue of the Acumen Journal with our *On-Screen Doc* profile.

The resulting *Results* dialog box is as pictured at right. We shall examine the four sections of this report.

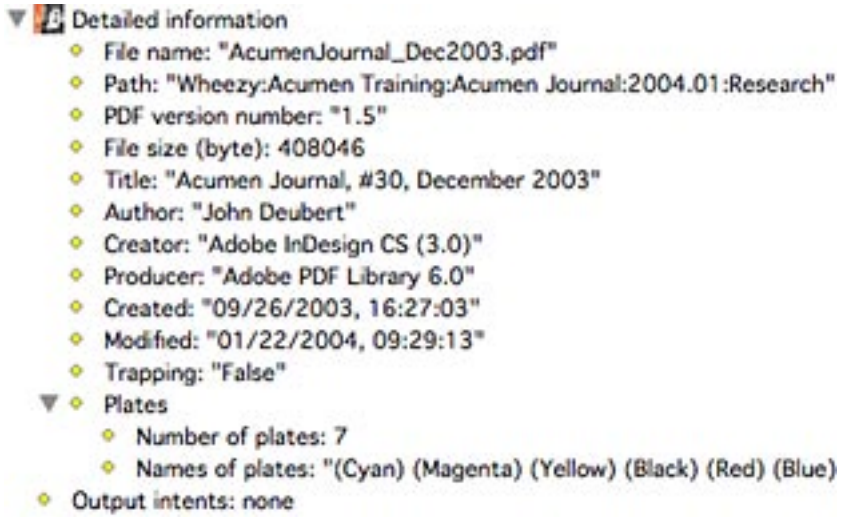


Problems

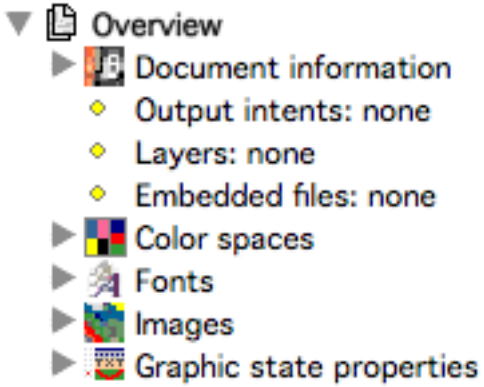
The "Problems" section lists items that failed some test specified in the preflight profile. In our case, the version of the file is greater than Acrobat 5 (that is, PDF version 1.4).

In some cases, you can double-click on an item in the Problems section and Acrobat will display the PDF page on which the problem occurs and highlight the troublesome item. This works only with problems associated with a specific item in the file, such as an image with too high a resolution.

The Problems section also has an entry labelled "Detailed Information." Surprisingly, if you click on the disclosure triangle (or the "+" sign in the Windows version) you find that this doesn't present details of the problems, but information about the document as a whole.



Overview The Overview section of the preflights results presents a document-wide summary of all the fonts, images, color spaces, etc. that occur in the PDF file.

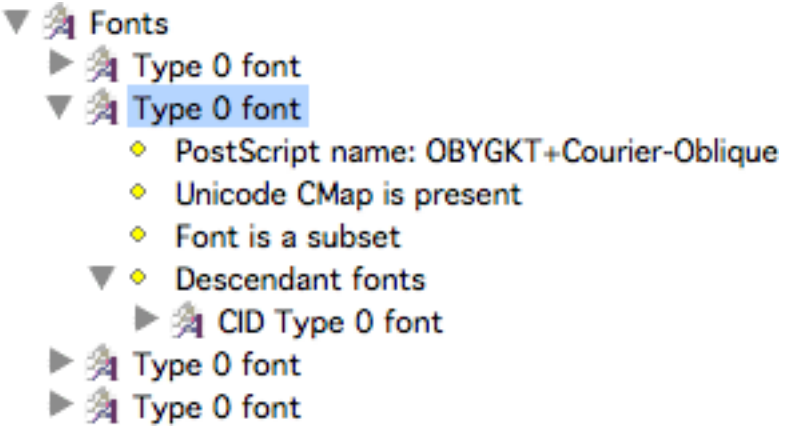


The first subtitle in this section, "Document information," heads up exactly the same information as appeared in the "Detailed information" section of *Problems;* when disclosed, the list of items looks exactly as in the illustration at above right.

The rest of the overview section lists all the resources used in this PDF file.

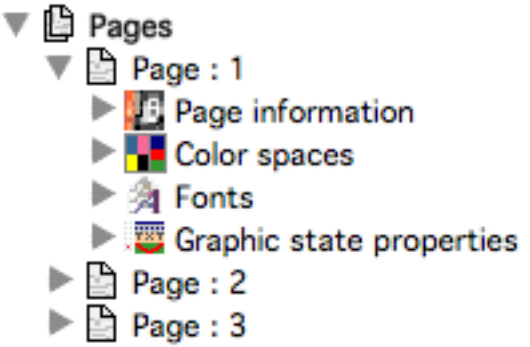
Next Page ->

As an example, if you expose the *Fonts* list, you will see a long list of the fonts that are used in the document. The list is a little disappointing at first, since it simply lists the font type ("Type 0 font", "TrueType font", etc.). To get the actual information about a font, you must click on its disclosure control.



Note that subsetted fonts in a PDF file have algorithmically-generated names that incorporate the name of the original font. (For example, the font in the report above, OBYGKT+Courier-Oblique, is a subsetted version of Courier-Oblique.)

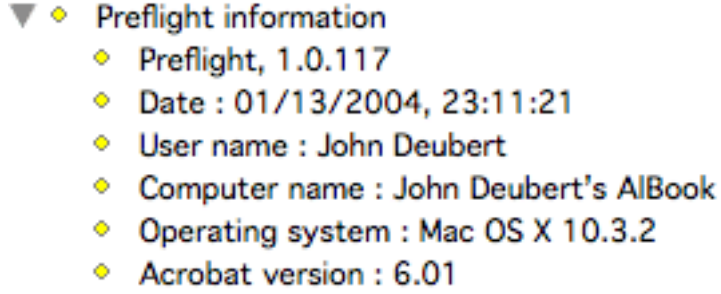
Pages The *Pages* section of the preflight result lists page-by-page information. This is the same information as is listed in the *Overview* section, but separated into individual pages.



Next Page ->

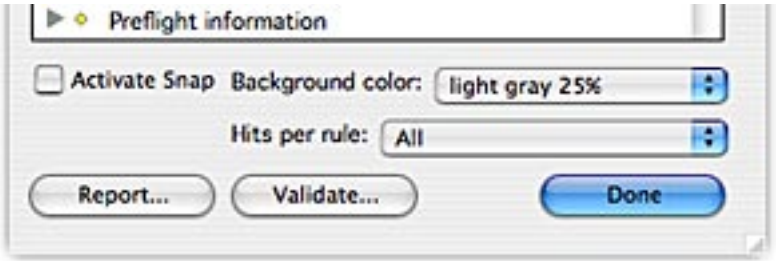
Preflight

Finally, the *Preflight* section of the results lists information about the preflight session itself: the date of the session, the computer on which the scan was performed, etc.



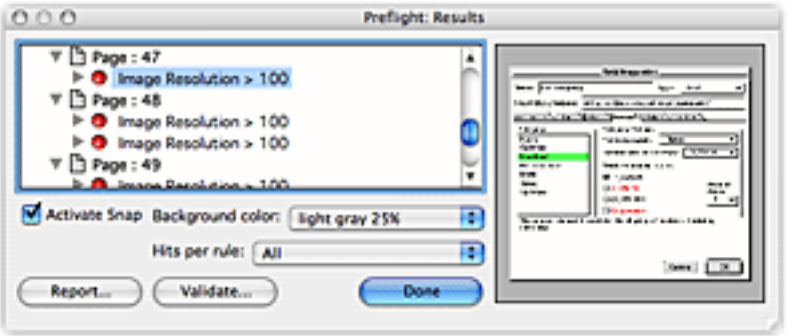
Other Report Options

The *Preflight Results* dialog box has a set of four controls at the bottom that give you access to some things you can do with the results.



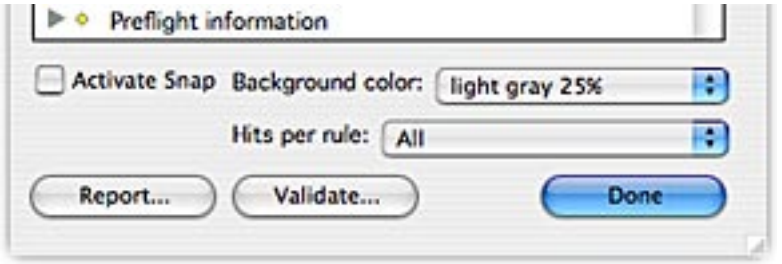
Snap

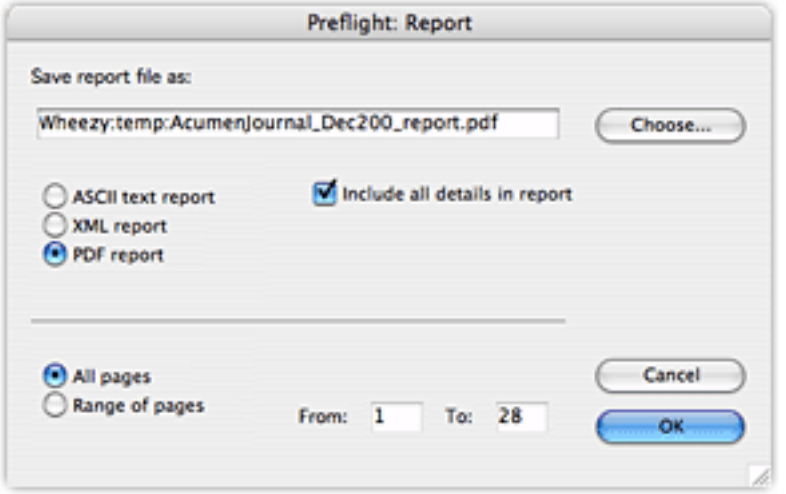
For example, if you select the check box labelled "Activate Snap," a new panel slides out to one side of the dialog box. This panel displays a picture of any image or graphic that you select in the list of results. This is very useful, because images and graphics are not otherwise labelled in the profile results list.



Next Page ->

Report The *Report* button tells Acrobat to save the preflight results to a file. When you click the button, you are presented with a dialog box that asks you what format you want for the report and where you would like it to be saved on your disk.





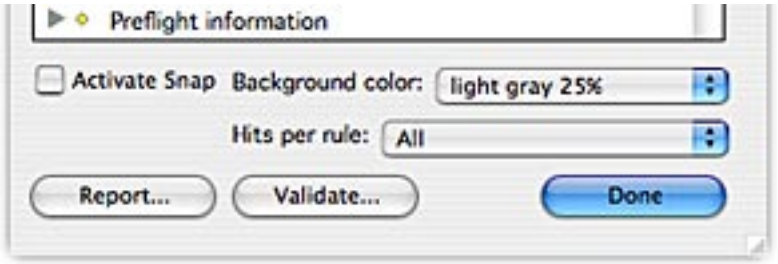
You may save the report in any of three formats:

- *PDF* - A PDF file containing the preflight results. Any problem items will be highlighted with a gray background, the actual shade of gray determined by the *Background Color* pop-up menu (upper right). Surprisingly, the actual preflight results are presented in this new file as a series of bookmarks.
- *XML* - This format could be used by a document manager to automatically check the readiness of your file for print.
- *ASCII text* - This presents the preflight result in a plain-vanilla text file.

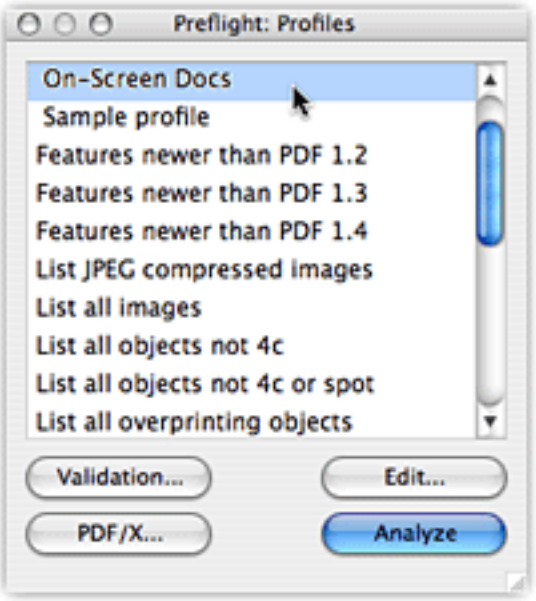
Next Page ->

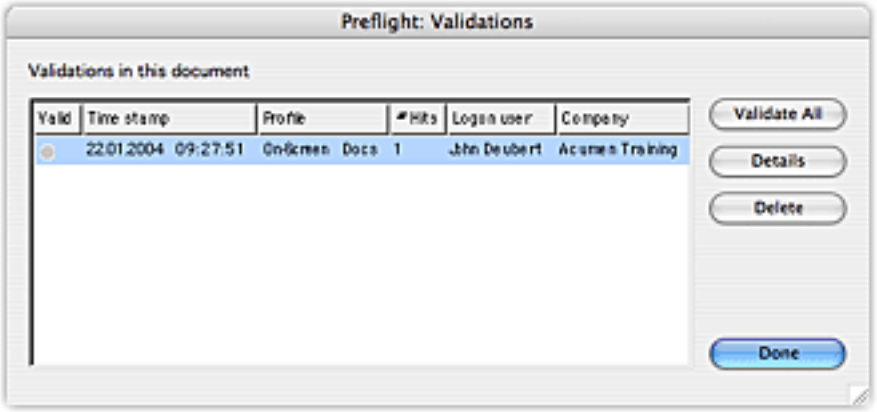
Acumen Journal: Acrobat User 9

Validate This final button in the Preflight Results dialog box tells Acrobat to embed the preflight result report in the PDF file, itself.



A person later opening your PDF file could bring up the *Preflight* dialog box (by selecting *Document>Preflight,* remember) and click on the *Validation* button (at right); Acrobat will present them with a list of all the preflight data stored in the file (below).





A person who double-clicks on a listed "validation" will be presented with a dialog box displaying the preflight information generated by the original scan.

Next Page ->

Acumen Journal: Acrobat User 10

PDF/X Finally, the Preflight Profiles dialog box lets you convert your PDF file to a PDF/X file (and check to see if it already is one).

What's PDF/X? A PDF/X file is a PDF file that obeys certain rules that ensure the file will work well in a pre-press environment. There are two sets of these rules: *PDF/X-1a* and *PDF/X-3*.

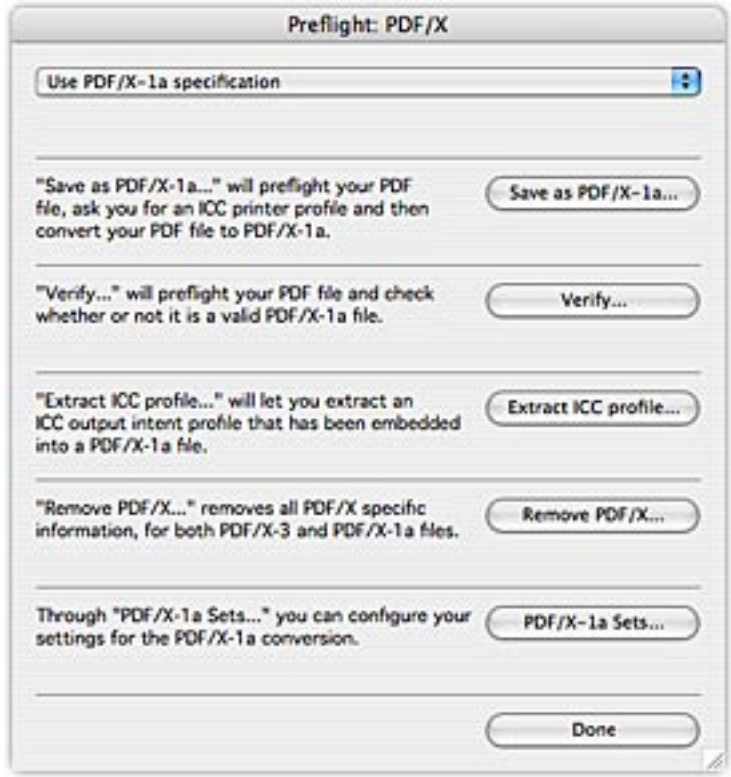
A file conforming to PDF/X-1a will obey these rules:

- The file will be self-contained. Thus all fonts, images, etc. will be embedded in the PDF file.
- The file contains a variety of information (a *TrimBox,* trapping, etc.) useful to documents printed on a press.
- Colors within the file will be CMYK or spot.

PDF/X-3 is the same as PDF/X-1a, except that colors may be in one of the CIE-based color spaces, allowing for color management.

PDF/X and Preflight If you click on the PDF/X button in the *Preflight Profiles* dialog box, Acrobat presents you with another dialog box that let's you check to see if your PDF file is compliant with PDF/X-1a.

Next Page ->



This dialog box has a pop-up menu at the top that lets you choose PDF/X-1a or -3.

The remainder of the dialog box is taken up by a set of buttons that perform the following actions on your PDF file:

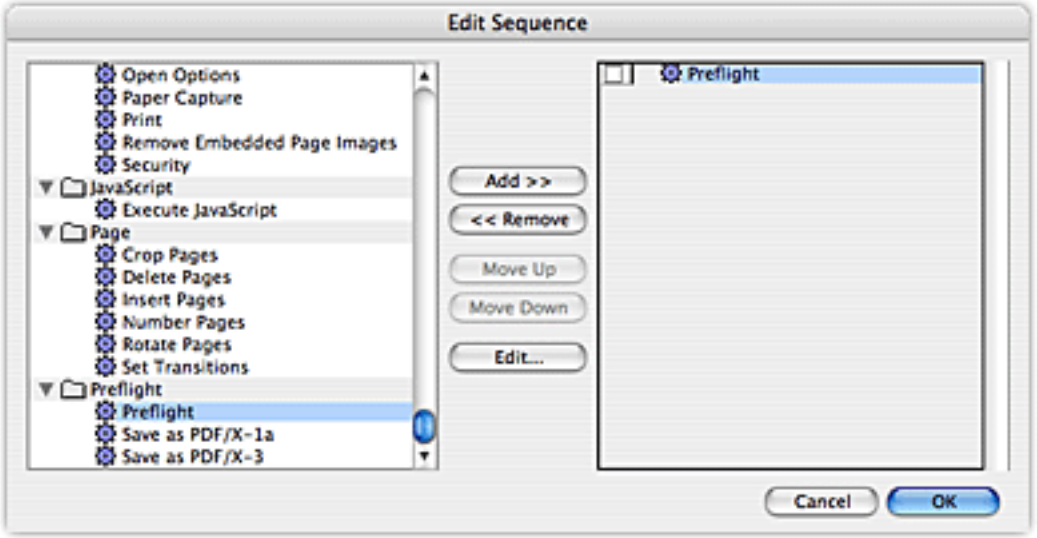
- *Save as PDF/X…* - Acrobat preflights your PDF file, making sure the colors are all correct, etc., and insert other information to make it a PDF/X file.
- *Verify…* - Acrobat runs a preflight scan on your PDF file, checking that it is appropriate for conversion to PDF/X.
- *Extract ICC Profile…* - PDF/X-3 files are required to have an ICC profile for the device to which they are intended to print. This button extracts the profile to a separate file.
- *Remove PDF/X* - Acrobat removes all of the PDF/X-specific information from the PDF file.
- *PDF/X Sets* - Save a set of scan settings as a named set for convenience later.

I'm being purposefully brief here. We'll come back to PDF/X in a later *Journal* article.

Next Page ->

Acumen Journal: Acrobat User 12

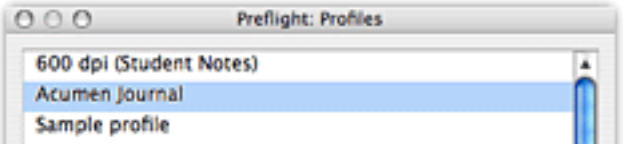
Batch Preflight



One last note before we're done: Acrobat 6's Batch Processing mechanism enables you to preflight batches of files. For more information, I refer you to the February 2003 issue of the *Journal*, which discusses batch processing in Acrobat 5; Acrobat 6 batch processing is pretty much the same as 5's, except the menu item by which you access it has been moved to *Advanced>Batch Processing*.

Next Month

Preflighting is a welcome addition to Acrobat 6's repertoire. However, it becomes most useful when you make your own preflight profiles, so that you can scan for those things in which you are particularly interested. (For example, I use home-grown profiles for the *journal,* my student notes, and my class overhead files.)



Next month, we shall see how to make our own profiles.

Return to Main Menu

Setpagedevice and Mysteriously Blank Pages

When PostScript was first designed, laser printers didn't offer much in the way of features. The original *LaserWriter,* for example, could select manual feed and paper size and that was about it. As a consequence, the PostScript's original design didn't pay much attention to addressing printer-specific features.

Whenever a printer manufacturer came up with a new feature (duplex printing, say), they and their Adobe representative would decide how that feature would be invoked on that printer. No effort was made to maintain consistency among different printers, so the PostScript code by which you turned on duplex printing varied from one printer to the next:

```
setduplex
setduplexmode
statusdict /duplexmode true put
```

If you guessed wrong, sending *setduplex* to a printer that wanted *setduplexmode,* you got an *undefined* error.

Adobe fixed this in PostScript Level 2. All printer-specific feature requests now go through a single operator, *setpagedevice*. This operator offers absolute consistency from one printer to the next.

It also introduced a usually-unexplained mystery to the PostScript world: pages that print unexpectedly blank for no discernible reason.

Next Page ->

Acumen Journal: PostScript Tech 14

Background: *setpagedevice*

The *setpagedevice* operator takes a dictionary from the operand stack:

```
<<  /Duplex true
    /PageSize [ 612 792 ]       % [ width height ]
>> setpagedevice
```

Each key-value pair within the dictionary requests a particular printer-specific feature. The above call to *setpagedevice* asks for duplex printing on American letter-size paper.

Consistency

What's significant is that the key-values pair that request a given feature is utterly constant across devices. The above call to *setpagedevice* will *always* get you duplex printing on letter-size paper if those features are available at all.

Flexibility

Feature request in *setpagedevice* are cumulative; you can make your requests in a single call to *setpagedevice,* as we did above, or in separate calls:

```
<< /Duplex true >> setpagedevice
<< /PageSize [ 612 792 ] >> setpagedevice
```

The order in which you request features is not significant.

Erases the Page

The *setpagedevice* operator erases the current page; you must make your feature requests before you start marking page. As we shall see, this is the source of the Mystery.

Next Page ->

Acumen Journal: PostScript Tech 15

Failed Requests

What if a feature request fails? What if your printer doesn't do duplex printing?

You can specify what should happen if a feature is missing by supplying a *Policies* dictionary in your call to *setpagedevice:*

```
<<  /Duplex true
    /PageSize [ 612 792 ]       % [ width height ]
    /Policies   <<
        /Duplex 0
        /PageSize 6
    >>
>> setpagedevice
```

Within a Policies dictionary, each key is the name of a feature and the associated value is a code number that indicates what should happen if that feature is not available.

The above call to *setpagedevice* says: "Print duplex on letter-size paper. If you don't have duplex printing, then invoke policy code *0;* if you don't have letter-size paper, then invoke policy code *6*."

Next Page ->>

Policy Codes There are three policy codes that may be applied to any feature request:

0	Generate a *configurationerror*. This feature is so important to the job that you don't want to print it if the feature is not available.
1	Ignore the request. If you don't have duplex, print simplex.
2	Do something printer-specific. Do something determined by the printer manufacturer. Typically, the printing device will notify the user and then wait for the user to fix the situation. ("Please turn on the duplexer.")

PageSize Policy Codes There are some additional Policy codes that apply to failed PageSize requests.

3	Use the nearest available paper size and scale the image to fit.
4	Use the next larger available paper size and scale the image to fit.
5	Use the nearest available paper size and do not scale the image.
6	Use the next larger available paper size and do not scale the image.
7	Continue to use the current paper size.

Next Page ->

Acumen Journal: PostScript Tech 17

Unexpected Blank Pages

Okay, now the Mystery:

Under certain circumstances, the use of *setpagedevice* can cause later calls to *grestore* or *restore* to completely erase the current page. Fixing this can be worth several hours of your life if you don't know what's going on.

The *Current Device*

Under the hood, the *setpagedevice* operator changes parameters in an internal PostScript data structure called the *current device*. The word "device" in PostScript refers to a target for rendering. The current device in PostScript is usually a *page device,* a rendering target representing the entire page. PostScript maintains several devices; for example, characters in a font are rendered into a *cache device*.

The *setpagedevice* operator sets parameters in the current page device structure. Whenever the current page device changes—for example, when *setpagedevice* modifies any of its parameters—PostScript reinitializes its buffer, erasing the page.

gsave, *grestore*, and *setpagedevice*

As it turns out, the current device is part of the graphics state and so is subject to *gsave* and *grestore*. If you call *setpagedevice* between a *gsave-grestore* pair, the *grestore* will reset the feature you modified with *setpagedevice*.

Next Page ->

Acumen Journal: PostScript Tech 18

Thus, in the following code:

```
gsave
/PageSize [ 612 1008 ]
...
grestore
```

the *grestore* will return the page size to whatever it was when we did the *gsave*.

Erasing the page

Less obviously, when *grestore* returns the graphics state to its earlier condition in the above code, it also resets the current device (part of the graphics state), which erases the current page.

Any time you call *setpagedevice* between *gsave* and *grestore,* the *grestore* will erase the page. This is the only circumstance I know of in which *grestore* can remove ink from the page. The same holds true of *save* and *restore,* which do an implicit *gsave* and *grestore*.

Next Page ->

Acumen Journal: PostScript Tech 19

Encapsulated PostScript

The circumstance in which you may bump into this involves the importing of EPS files into your PostScript.

When an application imports an EPS file, at print time it must place a *save/restore* pair around the file's PostScript code (so as to recover the memory used by the EPS code and discard its key-value pairs).

```
/ostate save def
...
... eps code goes here
...
ostate restore
```

Since, *save* and *restore* save and restore the graphics state, if there are any calls to *setpagedevice* in the EPS file's PostScript code, the *restore* will erase the page.

Properly, an EPS file should never make calls to *setpagedevice*. As an illustration format, EPS has no business trying to set page size or otherwise messing around with printer-specific features.

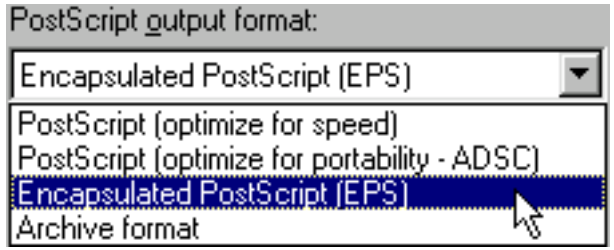
Enter Microsoft Windows.

Next Page ->

Acumen Journal: PostScript Tech 20

Printing to EPS

In Windows, when you print to a file, you are given the option of printing to an Encapsulated PostScript file. Unfortunately, in at least some versions of Windows (I haven't checked them all), the resulting EPS file contains calls to *setpagedevice*.



An application that imports one of these Windows print-to-EPS files will put a *save* and *restore* around it at print time; the *restore* will erase the page because of the *setpagedevice* in the EPS file.

Fixing It

So, what to do?

The fix is simple: somewhere after the pre-EPS *save,* redefine *setpagedevice* to do a *pop:*

```
/ostate save def
/setpagedevice /pop load def
...
...
ostate restore
```

Now, when the EPS code executes *setpagedevice,* it simple throws away the dictionary argument, rather than actually changing the current device. Just as applications that import EPS files routinely redefine *showpage* to an empty procedure, they should also always redefine *setpagedevice* to *pop*.

Next Page ->

Acumen Journal: PostScript Tech 21

It's Pretty Rare, Really

You won't bump into this problem very often. Most people make EPS files by exporting directly from within a professional-grade application, such as *Adobe Illustrator*. Only rarely will you encounter a file made by printing to EPS.

Still, when you *do* encounter one, it can take a frustrating amount of time to figure out what's going on.

Return to Main Menu

Schedule of Classes, Feb - May 2004

Following are the dates of Acumen Training's upcoming PostScript and PDF Technical classes. Clicking on a class name below will take you to the description of that class on the Acumen training website.

These classes are taught in Orange County, California and on corporate sites world-wide. See the Acumen Training web site for more information.

Technical Classes

PDF File Content and Structure	Feb 9–12		Jun 14–17
PostScript Foundations			Apr 26–30
Variable Data PostScript			
Advanced PostScript			
PostScript for Support Engineers		Mar 15–19	
Jaws Development		*On-site only*	

Course Fee

The PostScript and PDF classes cost $2,000 per student.

Registration Info

Acrobat Classes

Acrobat Class Schedule

These classes are taught occasionally in Costa Mesa, California, and on corporate sites. Clicking on a course name below will take you to the class description on the Acumen Training web site.

Acrobat Essentials

Interactive Acrobat

Creating Acrobat Forms

No Acrobat classes scheduled for this quarter. See the Acumen Training website regarding setting up an on-site class.

Acrobat Class Fees

Acrobat Essentials and *Creating Acrobat Forms* (½-day each) cost $180.00 or $340.00 for both classes. *Troubleshooting With PitStop* (full day) is $340.00. In all cases, there is a 10% discount if three or more people from the same organization sign up for the same class.

Registration ->

Return to Main Menu

Contacting John Deubert at Acumen Training

For more information

For class descriptions, on-site arrangements or any other information about Acumen's classes:

Web site: http://www.acumentraining.com **email:** john@acumentraining.com

telephone: 949-248-1241

mail: 24996 Danamaple, Dana Point, CA 92629

Registering for Classes

To register for an Acumen Training class, contact John any of the following ways:

Register On-line: http://www.acumentraining.com/registration.html

email: registration@acumentraining.com

telephone: 949-248-1241

mail: 24996 Danamaple, Dana Point, CA 92629

Back issues

Back issues of the *Acumen Journal* are available at the Acumen Training website:
http://www.acumenjournal.com/AcumenJournal.html

Return to First Page

What's New at Acumen Training?

New *Journal* Fonts

I've switched to new fonts in the *Acumen Journal*. The new fonts are all OpenType, letting me use some of the additional features of that format. Anyone hate the new look?

Oh, yes. I also changed from *QuarkXpress 6.0* to *InDesign CS*.

Anyone Interested in Acrobat JavaScript?

I'm contemplating developing a pair of two-day classes in programming Acrobat in JavaScript.

The first class would be a beginner's class, presuming no programming experience. It would introduce the student to JavaScript in the context of writing programs for Acrobat. It would require that the student have good Acrobat skills, but that's about it. We'll look at a variety of tasks you can carry out in Acrobat using JavaScript.

The second class would build on the first, introducing new programming concepts and discuss such things as interacting with databases, etc.

Currently I'm trying to get an idea as to how much interest there would be in such a class. If you would be interested, please drop me a note at john@acumentraining.com. There's no commitment and I won't bother you when the class is available (unless you ask me to, of course). I'm just trying to get a feel as to whether such a class would be worth doing.

Return to First Page

Journal Feedback

If you have any comments regarding the *Acumen Journal,* please let me know. In particular, I am looking for three types of information:

Comments on usefulness. Does the Journal provide you with worthwhile information? Was it well written and understandable? Do you like it, hate it? Does it make your lobotomy unnoticeable?

Suggestions for articles. Each Journal issue contains one article each on PostScript and Acrobat. What topics would you like me to write about?

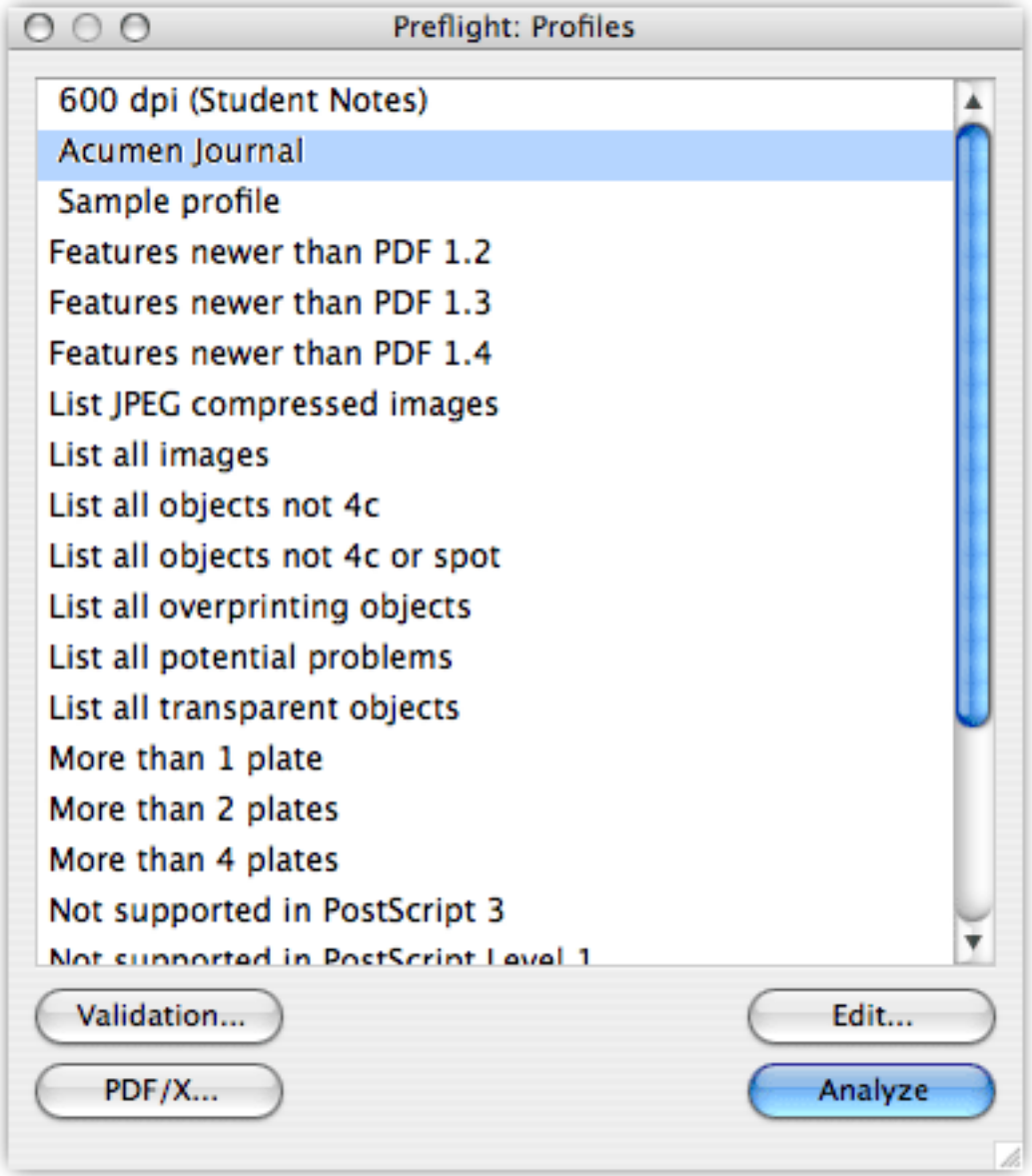
Questions and Answers. Do you have any questions about Acrobat, PDF or PostScript? Feel free to email me about. I'll answer your question if I can. (If enough people ask the same question, I can turn it into a Journal article.)

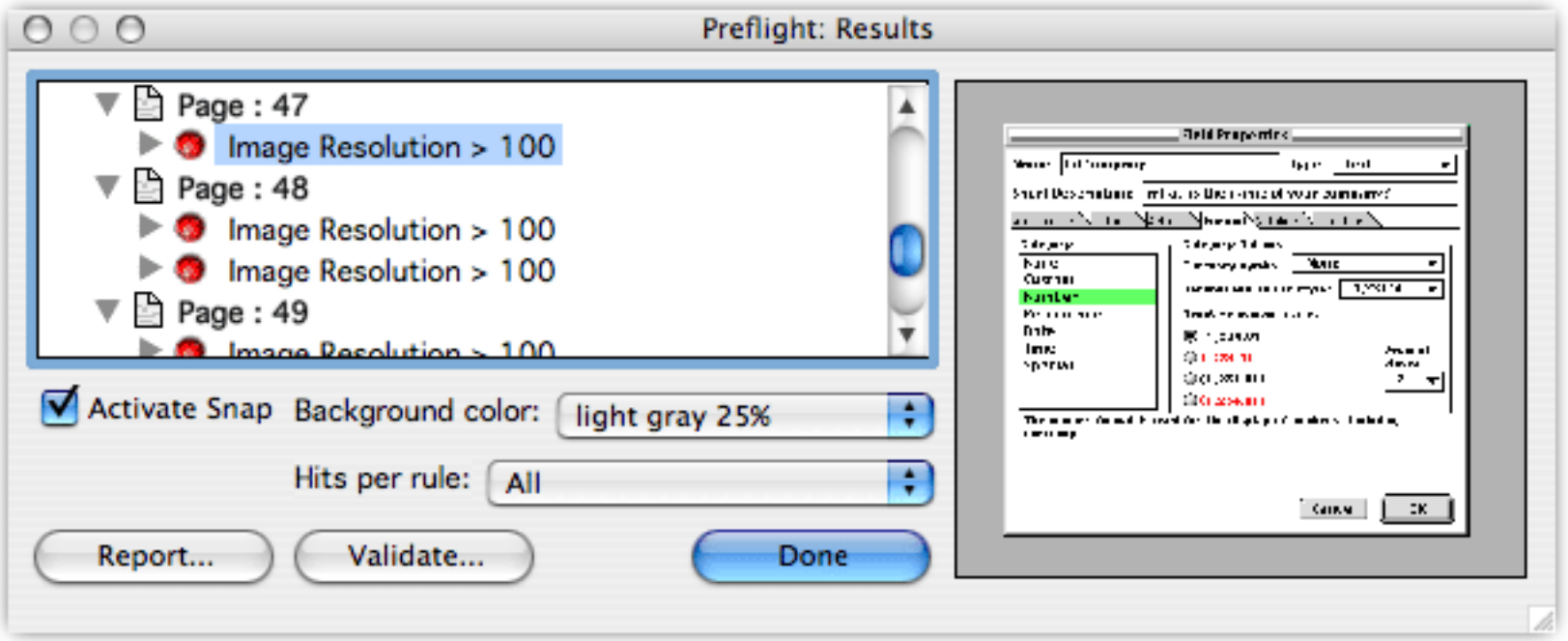
Please send any comments, questions, or problems to:

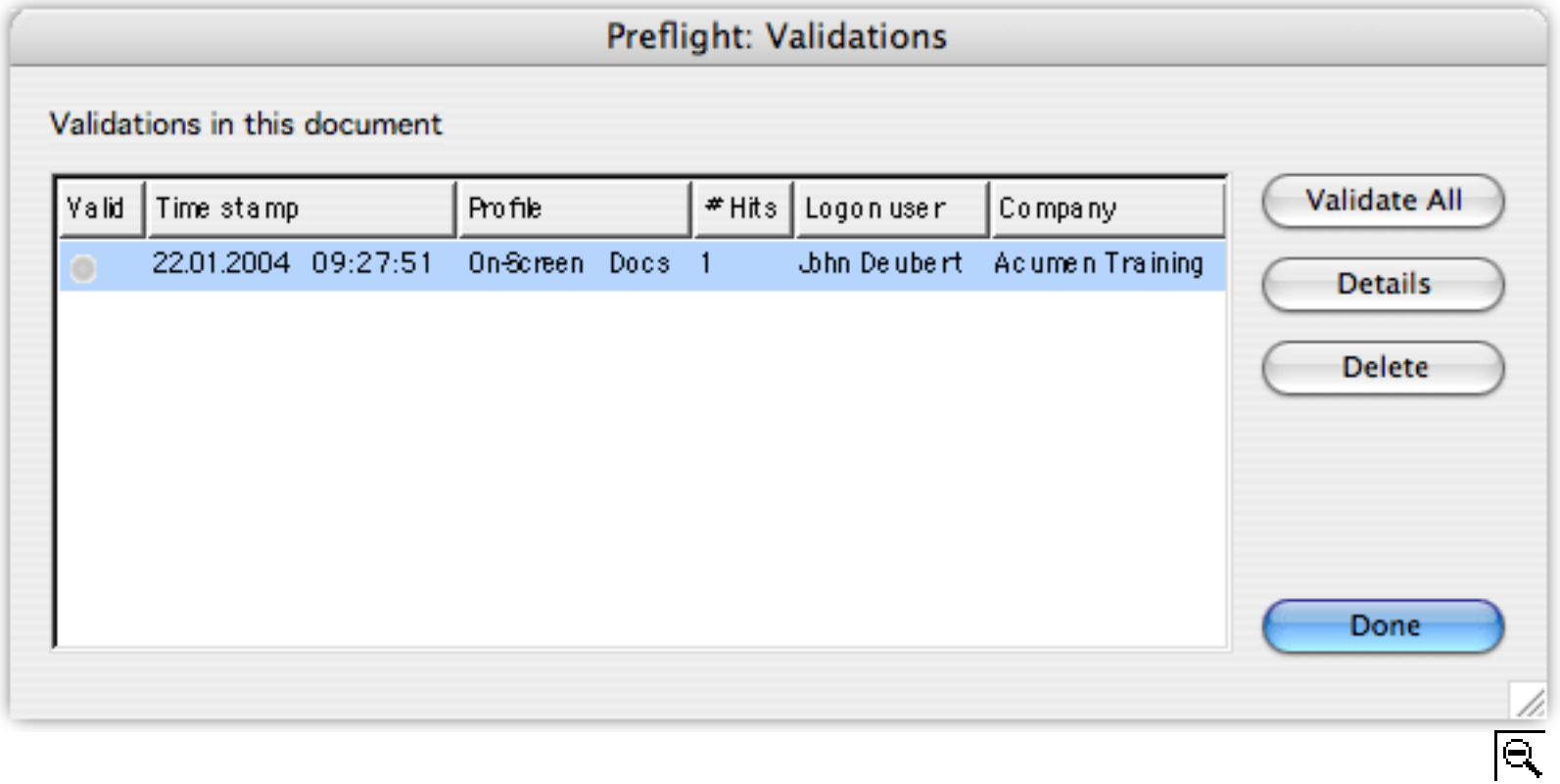
journal@acumentraining.com

Return to Menu

Preflight Profiles




Preflight: Results
Page : 47
Image Resolution > 100
Page : 48
Image Resolution > 100
Image Resolution > 100
Page : 49
Activate Snap
Background color:
light gray 25%
Hits per rule:
All
Report...
Validate...
Done


Preflight: Validations
Validations in this document
Valid
Time stamp
Profile
Hits
Logon user
Company
22.01.2004 09:27:51
On-Screen Docs
1
John Deubert
Acumen Training
Validate All
Details
Delete
Done