

## Table of Contents

### [The Acrobat User](#)

#### **Acrobat "WebLinks"**

Adobe Acrobat can automatically scan your PDF file and add links to web locations anywhere a recognizable URL appears in text.

### [PostScript Tech](#)

#### **Early Name Lookup With //Double-Slash**

PostScript has a little-used double-slash delimiter that lets you do define-time name lookup of constants within procedures. This month we'll see how to use this to reduce name lookup at execution time.

### [Class Schedule](#)

#### **March-April-May**

Where and when are we teaching our Acrobat and PostScript classes? See here!

### [What's New?](#)

#### **See you at Seybold?**

Going to be at February's Seybold Seminars in New York? Look me up!

### [Contacting Acumen](#)

Telephone number, email address, postal address, all the ways of getting to Acumen.

[Journal feedback: suggestions for articles, questions, etc.](#)

#### **By the way...**

This month's Journal articles are shorter than usual. I'm working on an all-consuming project through mid-March.

*JD*

# Acrobat “WebLinks”

One of the nicer abilities of Acrobat is to establish a link between a PDF file and a site on the World Wide Web. A link in a PDF file, like [this one](#), can launch your web browser and take you to a web page.

Alternatively, the link could tell Acrobat 5 to capture the web page and convert it to a PDF page.

Either way, it makes an impressive integration of your PDF document with the World Wide Web.

If you have a long PDF document with many URLs you want to turn into links, adding the links one at a time can be unbelievably tedious.

Happily, under certain circumstances, Acrobat will do it for you. If you use Acrobat’s “WebLinks” feature, Acrobat will scan through your PDF file and place an appropriate link over any URLs in the text.

It’s fast and very easy.

Let’s see how to use it.

[Next Page ->](#)

## How it Works

When you invoke the *Create WebLinks* feature, Acrobat searches through the your PDF file, looking for text that starts with "http://", as at right.

Whenever Acrobat finds a recognizable URL, it creates a link to the specified URL and places it on top of the text. If you later click on the text, Acrobat will launch your web browser and take you to that site. Alternatively, you can have Acrobat automatically use its "web capture" ability to convert the web page to PDF and open it in Acrobat.

<http://www.planetpdf.com>

<http://www.pdfzone.com>

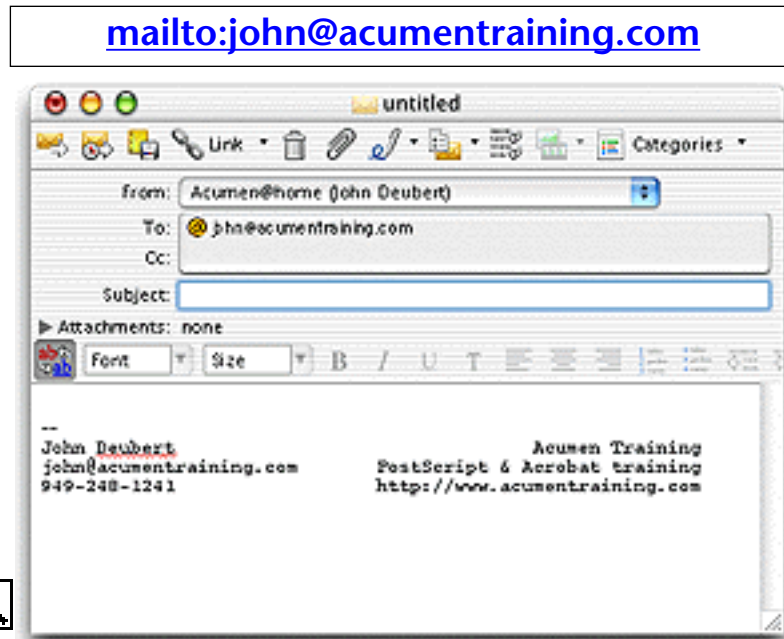
<http://www.adobe.com>

### **Mailto: Links**

Acrobat will also recognize text that starts with "mailto:". The resulting link will launch your mail client and present you with a blank email form already made out to the address that follows "mailto."

<mailto:john@acumentraining.com>

[Next Page ->](#)



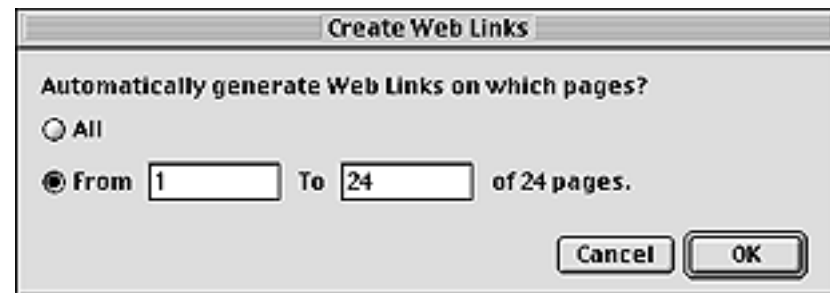
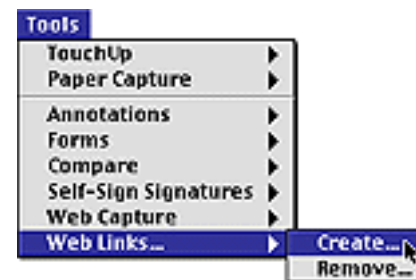
## Creating WebLinks

You tell Acrobat to scan through your document and create WebLinks by selecting (deep breath, now) *Tools>Locate Web Addresses>Create WebLinks from URLs in text*.

In Acrobat 4, this menu items is a bit more tersely worded. You select *Tools>WebLinks>Create*. This is less clear, perhaps, but makes for a narrower sub-menu. (That may not be a valid software design goal, I admit.)

Acrobat will ask you what range of pages you want it to scan. When you click "OK," it runs through the Acrobat file and creates all the links.

[Next Page ->](#)

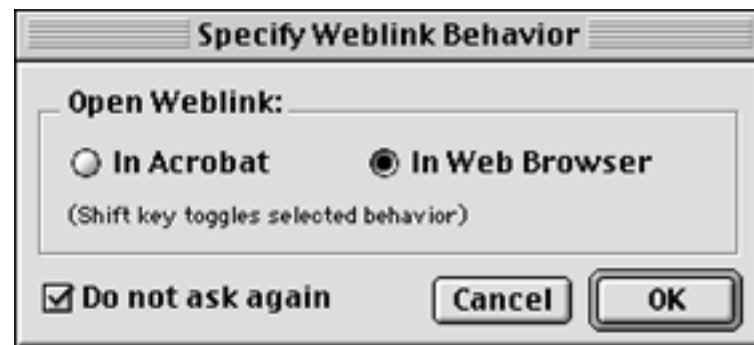


**Type of Weblink** When scanning your file, Acrobat may ask you what you want to happen when you click on the links created in the process.

Your choices are:

*Open Weblink in Acrobat* Acrobat will convert the web page to PDF and open it in Acrobat.

*Open Weblink in Web Browser* Acrobat will open your system's default web browser and go to the specified URL.



[Next Page ->](#)

## A Couple of Comments

### No Duplicate Links

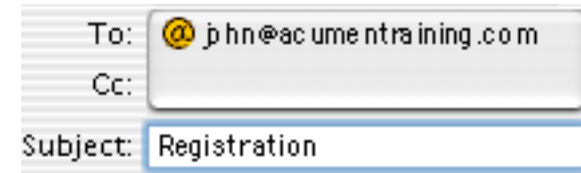
If Acrobat finds a URL that already has a link, it doesn't add a new link on top of the old. This prevents you from piling up several links in the same place.

### Passing parameters to **mailto:**

It doesn't happen much, but the text for your *mailto:* link can contain parameters, such as:

*mailto:john@acumentraining.com?subject=Registration*

This will give you a blank email form with the address and subject both filled out.

A screenshot of an email client's 'Compose' window. The 'To:' field contains an email icon followed by 'john@acumentraining.com'. The 'Cc:' field is empty. The 'Subject:' field contains the text 'Registration'. The fields are set against a light blue background with horizontal lines.

However, your subject and other parameters will end at the first whitespace character Acrobat encounters.

This means multi-word subjects, for example, will be truncated at the first word.

### Kerned URLs

If the text in a URL name has been kerned, WebLinks may incorrectly parse the URL, yielding an incorrect link. You may want to quickly page through your PDF file with the link tool selected (which makes links visible), looking for links that don't match the size on the page the corresponding URL.

[Go to Main Menu](#)

# Reducing Name Lookup With //Double-Slash

Nearly every PostScript programmer is familiar with the use of the *bind* operator. It takes a procedure body and replaces all executable operator names within the procedure with their definitions.

That is, in the following PostScript code...

```
/ptsPerInch 72 def
/inch { ptsPerInch mul } bind def
```

...the *inch* procedure ends up with the name *ptsPerInch* and the operator object for *mul* (symbolize below as “—mul—”).

```
{ ptsPerInch —mul— }
```

When *inch* is executed, there is no name lookup associated with *mul*, since the name has already been replaced by its definition.

The interpreter *will* have to look up *ptsPerInch*, however. This is too bad, since *ptsPerInch* is a constant; looking it up always returns 72.

The double-slash (//) is a *very* little-used Postscript delimiter that may be applied to names within a procedure body; it tells the PostScript scanner to look up the name and place the value associated with that name into the procedure.

Double-slash is as useful as *bind*, and yet almost no one uses it.

This month I intend to fix that.

[Next Page ->](#)

## Using Double-Slash

If you have taken the *PostScript Foundations* class, you may remember the example that printed text between specified left and right margins. The example starts out by defining three constants and a *newline* procedure:

```
/LM 72 def
/RM 216 def
/LineHt 14 def
```

```
/newline { LM currentpoint LineHt sub exch pop moveto } bind def
```

*Newline* moves the current point to the beginning of the next line. The *bind* operator replaces all of the operator names with their definitions, leaving only *LM* and *LineHt* to be looked up when we execute *newline*.

However, neither *LM* nor *LineHt* change values over the course of the PostScript example. They can be double-slashed, therefore, and their values, rather than the names, will be placed in the *newline* procedure.

```
/newline { //LM currentpoint //LineHt sub exch pop moveto } bind def
```

*Newline* now contains:

```
{ 72 -currentpoint- 14 -sub- -exch- -pop- -moveto- }
```

Executing *newline* now entails no name lookup at all, except for the lookup of “newline,” itself!

[Next Page ->](#)



## When to use it

**Constants** Double-slash is mostly used with the names of constants within a procedure, as in *newline*. Any name within a procedure whose value does not change over the duration of the PostScript program may be double-slashed.

Somewhat counter-intuitively, this includes strings that are being used as buffers:

```
/buffer 1000 string def
/ReadALineOfText { currentfile //buffer readline } bind def
```

At first glance, *buffer* looks like a variable, rather than a constant, since it has different text each time *ReadALineOfText* returns.

On second glance, however, *buffer* is actually a constant string; that is, a constant string object pointing to a constant place in VM. Different text is put into that string each time you call the procedure, but the string object itself doesn't change.

[Next Page ->](#)

## Not with procedure names

By and large, you cannot usefully double-slash procedure names. The following will not do what you want:

```
/PrintWord          % (str) => ---
{    dup stringwidth pop
    currentpoint pop add
    //RM gt { //newline} if
    show
} bind def
```

The line with the double-slashes would become:

```
216 -gt- { {72 -currentpoint- 14 -sub- -exch- -pop- -moveto-} } -if-
```

Note that the procedure handed to *if*, when executed, simply places the inner procedure on the stack; it never executes the 72-currentpoint-etc.

So, don't double-slash procedure names. It's not useful.

[Next Page ->](#)

**Well, OK, *sometimes*  
with procedure names**

There is one case where it *is* useful to double-slash procedure names. Consider our *PrintWord* definition again, with a change:

```
/PrintWord      % (str) => ---
{    dup stringwidth pop
    currentpoint pop add
    //RM gt //newline if
    show
} bind def
```

The rather odd-looking *//newline* places the procedure body associated with “newline” into the *PrintWord* procedure before the *if*. That is, the line with the double-slashes becomes:

```
216 -gt- { 72 -currentpoint- 14 -sub- -exch- -pop- -moveto- } -if-
```

What we’ve done is eliminated the name lookup of “newline” and a little associated bookkeeping associated with this name lookup.

Anytime a procedure argument to *if*, *ifelse*, *repeat*, etc. contains only a single procedure name, you can replace the procedure with the double-slashed name.

[Next Page ->](#)

Thus, the loop:

```
[ (This ) (is ) (printed ) (between ) (the ) (margins.) ]  
{ PrintWord } forall
```

can become

```
[ (This ) (is ) (printed ) (between ) (the ) (margins.) ]  
//PrintWord forall
```

**So use it!** Eliminating name lookup for constants won't quadruple your printing speed, truth be told. Still, if efficiency is a concern (and when is it not?), why spend time repeatedly looking up names at execution time that could be looked up only once when you define your procedures?

Double-slash those constants!

Besides, a line of PostScript code like

```
//RM gt //newline if
```

is bound to impress at least *some* of your friends.

[Go to Main Menu](#)

# Schedule of Classes, March – May, 2002

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

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

## PostScript Classes

[PostScript Foundations](#)      March 18 - 22                      May 13 - 17

[Advanced PostScript](#)      April 29 – May 3

[PostScript for Support  
Engineers](#)      April 15 - 19

[Jaws Development](#)      April 2 - 5

For more classes, go to [www.acumentraining.com/schedule.html](http://www.acumentraining.com/schedule.html)

**PostScript Course Fees**      PostScript classes cost \$2,000 per student.  
These classes may also be taught on your organization's site.  
Go to [www.acumentraining.com/onsite.html](http://www.acumentraining.com/onsite.html) for more information.

[Registration →](#)  
[Acrobat Classes →](#)

# Acrobat Class Schedule

**On-Site Only** These classes are taught only on corporate sites. If you have an interest in any of these classes for your group, please see the Acumen Training website regarding arranging an on-site class.

**[Acrobat Essentials](#)** This class teaches the student how to make perfect PDF files. It includes complete coverage of the meaning and proper settings of all of the Distiller Job Options.

**[Interactive Acrobat](#)** Here we show you how to add bookmarks, links, buttons, sounds, movies, form fields, and other interactive features to an Acrobat file.

**[Creating Acrobat Forms](#)** This class shows you how to make interactive forms in Adobe Acrobat. It steps you through creating the form, posting form contents to a server, and everything else you need to create a working PDF form.

**[Troubleshooting with  
Enfocus' PitStop](#)** This class shows the student how to use all of the capabilities of this popular editing and preflight software.

[Back to PostScript Classes](#)

[Return to First Page](#)

# 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](mailto:john@acumentraining.com)

**telephone:** 949-248-1241

**mail:** 25142 Danalaurel, 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](mailto:registration@acumentraining.com)

**telephone:** 949-248-1241

**mail:** 25142 Danalaurel, Dana Point, CA 92629

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

[Return to First Page](#)

# What's New at Acumen Training?

## See you at Seybold

I'll be teaching two Acrobat classes at the New York Seybold Seminars, February 19 & 20. These are:

### **PDF for Prepress**

A slightly shortened version of *Acrobat Essentials*.

### **Creating Acrobat Forms**

A slightly shortened version of my usual *Creating Acrobat Forms* class.

If you attend the Seybold Seminars in February, come by and say "Hi."

[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 you want to disappear into the wilderness for years?

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

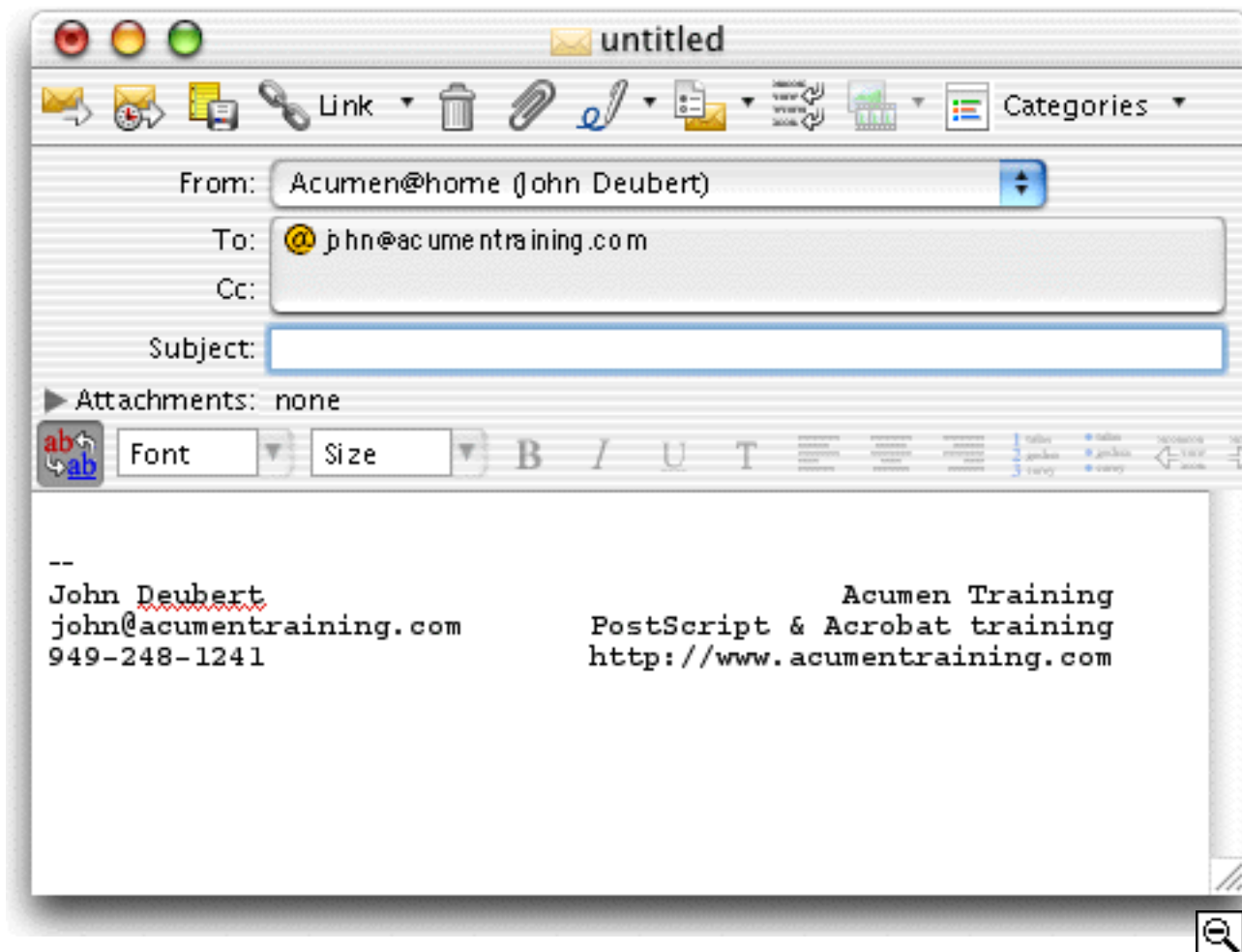
**Questions and Answers.** Do you have any questions about Acrobat, PDF or PostScript? Feel free to email me about. I'll answer your question if I can. If enough people ask the same question, I can turn it into a Journal article.

Please send any comments, questions, or problems to:

[journal@acumentraining.com](mailto:journal@acumentraining.com)

[Return to Menu](#)

## Pre-addressed Email Form



The screenshot shows a window titled "untitled" with a toolbar containing icons for sending, scheduling, attachments, links, deleting, inserting, and categories. The email fields are as follows:

From: Acumen@home (John Deubert)

To: @jhn@acumentraining.com

Cc:

Subject:

Attachments: none

Below the attachments section is a rich text editor toolbar with options for font, size, bold, italic, underline, text color, background color, bulleted list, numbered list, link, unlink, and undo. The email body contains the following text:

--  
John Deubert  
john@acumentraining.com  
949-248-1241

Acumen Training  
PostScript & Acrobat training  
<http://www.acumentraining.com>