by Don Lancaster

# PDF Links and PDF Link Checking

One of the many compelling advantages to online content over print media is the ability to instantly *link* to other material. As internal to your present page; to related local pages; to other docs; or off site. Links can give you more detail, simpler tutorials, or advanced topics to explore. Besides routing you to other web pages, links can play sound clips, run animation, show movies, dial a phone, work with forms, or even turn on a hot tub.

Links are central to the HTML web language. Much less obvious is how and where you can use links in Acrobat PDF files. It does turn out that you can easily do as much and sometimes lots more with Acrobat links as you can with HTML. And often more conveniently and more attractively.

Once these few fundamentals are understood...

## Creating Acrobat PDF Links

Your foremost starting point in understanding Acrobat *PDF* links is Adobe Technical Note #5150 on their *PDFMark* operator. This doc is gotten from...

*http://www.adobe.com/suportservice/de vrelations/PDFS/TN5150.PDFMARK.PDF*

In general, various PDF mark operators add special new commands. These marking operators give you additional features or control abilities. Some examples include notes, links, bookmarks, forms, and articles.

The marking operators can get placed into your original PostScript sourcecode, or else may be added to an existing PDF file by a later editing process.

Your key PDFMark operator for external linking is */ANN* with a */URI* subtype. Here is the text that might be inserted into your sourcecode to link cururlname to a selected action rectangle on the page...

```
mark                           % start of data

/Rect [llx lly urx ury]        % box size & pos
/Border [0 0 0]                % border style
/Color  [0.7 0 0]              % border color
/Action <</Subtype /URI        % url link wanted
    /URI cururlname >>
/ANN                           % action desired

pdfmark                        % action command
```

There are several ways to create PDF links. These include using Acrobat Exchange, using some sourcecode *Document Generator*, or using my *Gonzo Utilities*...

***Using Exchange–*** Links are easily (but manually) created after PDF file generation by using Acrobat Exchange.

You select *tools*, then *link*. You then move the crosshairs to the lower left of your selection box. Typically, this will be slightly larger than the text to be linked. Click and drag to upper right to select your hot area. Then select your choices of visible or invisible; thin, medium, or thick; solid or dashed. At that point, you have a dozen options of what your link is to do. More often than not, you'll pick *world wide web link* for external links or *go to view* for internal ones. Use the *Edit URL* box to enter where you want to go.

Then click on *set link*.

There are limitations to Exchange created PDF links. The first is all the intensive time and labor involved when you need dozens to hundreds of links. The second is that your link locations are *separate* from your link text. This means that *any editing may cause the linking text to move elsewhere from its related link box.* Worse yet, all of your links might have to get manually re-entered for even the simplest of editing modifications.

***Using a Document Generator–*** It is ideal to embed your links directly into your original sourcecode document as it is generated. This gets tricky for several reasons. First, your editor or creation package has to understand PDFMarks and how they work. Second, some method has to exist to auto generate a hot selection box which matches the length and the font size of your intended linking text.

This can be especially messy if your linking text is split over two or more justified lines. Or, worse, over a column split. Finally, some convenient "list" method should exist to pull your url locations up in managable blocks, rather than having to manually enter them.

Useful document generators having PDF link capability are starting to appear. I can't recommend the only one I've tested so far. But better ones should be here shortly.

***Using my Gonzo Utilities–*** I had a personal need for lots of effective PDF links, so I modified my *Gonzo Utilities* to semi-automatically place Acrobat links directly into all my sourcecode. The selected regions and the underlying text autosize and automatically track each other on further edits. Web url's are easily inserted as complete lists.

My Gonzo Utilities and their extensions do work on all platforms. These are also available free for your personal use. But Gonzo is *not* WYSIWIG, and the documentation does remain rather limited.

I use it because it does my stuff my way for me.

The Gonzo Utilities are found at...

http://www.tinaja.com/psutils/gonzo.ps

... while detailed linking examples are found in any of the ".PSL" files at *www.tinaja.com/muse01.html*. One typical example is *www.tinaja.com/glib/muse135.psl*

Here's how my Acrobat PDF url inserter works: Before your linking text, you enter a *surl* or "start url" command.

After your linking text, you enter the a name that links your desired url. Such as */netscape* or perhaps */ng3* for "newsgroup 3". Elsewhere in your code, you link the name and the url. Typically in a self-generating, self-defining dictionary, such as...

```
<<
    /adobe (http://www.adobe.com)
    /netscape (http://www.netscape.com)
        ...       ...       ...
    /parallax (http://www.parallaxinc.com)
    /tinaja (http://www.tinaja.com)
>>
    {mark exch /eurl cvx ] cvx def} forall
```

That last line converts everything in the dictionary into individual action definitions. The */eurl* operator then will generate the linking code when called.

Since there is no harm in having extra url's defined but not used, you block load all often used links at once.

As your actual text line gets justified, the Gonzo Utilities interpret the */surl* command. A guess is made as to how big the current typography is, and a box gets created that has a wider border around it. Your text color also gets changed. I prefer to use blue text, no underline, with an invisible hot box. After justification, the box size gets properly entered into the needed PDFMark code.

There are two minor gotchas. When your link title words overflow into the next justified line, you have to enter the second line text separately. And if your link title starts off flush left on the column, you may have to purposely add a few ignored leading spaces to make certain the */surl* gets activated on the right line.

## Internal PDF Linking

It is very easy to link to most any portion of most any HTML page. This can get done by adding a trailer of form *#gohere* to the end of the selected web url. A *target* named *gohere* is placed at an appropriate markng position. Linking to a selected point inside your *current* PDF document can be easily done using the *Go to view* Exchange tool.

Linking to a selected point inside another PDF or HTML doc is tricky. While the *#gohere* format is the same, this only works under certain circumstances. First, observe that there are two different ways of using Acrobat on line. With the *plug-in* method, code is placed internal to *Netscape* or another web browser that causes the browser to *internally* generate appropriate Acrobat output. But with the *helper* method, Acrobat gets activated in a separate window as a stand-alone application.

*Internal linking only works with the plug in method!*

I've got a linking demo at *www.tinaja.com/linkpdf1.html*

This one gets you back and forth between a HTML page and any of four internal locations in a PDF document. The demo definitely works with *Netscape* whenever an Acrobat Reader or Exchange module is used as a Netscape plug in. It *should* work ok on later versions of IE *if* the Adobe *ActiveX* patch is properly installed.

The demo will *not* work if helper apps are in use or if Acrobat plug-ins are not properly installed.

## PDF Link Checking

Very few things are more infuriating to your users than a link that does not go anywhere. Or one which goes to a wildly wrong location. It is super important to personally verify that *all* your links are valid at upload time. And then recheck them every now and then for currency.

Your obvious thing to do is to sit down and carefully check each link by hand. But this can get rather painful if dozens of links are involved. Or when several iterations are needed to pin down typos and relocations.

One rather lazy way to check for at least some *internal* HTML link errors is to view the referrals in your log file. A 404 error might point to a wrong internal url. At the same time, the referral info on the same log line tells you exactly where your error came from. It pays to always check your raw log files. Especially the referrals. But this is more useful as a second line of defense.

More on log file use in *www.tinaja.com/acrob01.html* and *www.tinaja.com/weblib01.html*. Especially files...

| | |
|---|---|
| BANNYEAR.PDF | BWHISTLE.PDF |
| GRABURLS.PS | REFLOG1.HTML |
| REFSUM1.HTML | MUSE116.PDF |
| MUSE124.PDF | PDFLINK.PDF |
| WEAVEWEB.PDF | WEBERRU2.PS |
| WEBLOGU2.PS | WEBSITAN.PS |

There is a rather nice automated and free way to test HTML links online. This involves *Dr. HTML* A commercial program that can check your entire HTML site all at once. Finding broken links, spelling errors, any coding problems, structure errors, and lots more.

They also have a free online demo checker which works with only one of your pages at a time. Check this one out at *www2.imagiware.com/RxHTML* Or use my "HTML" button at *www.tinaja.com*

Sadly, Dr. HTML is not yet capable of testing links that are *inside of* PDF files. And I know of no other service or any software routine that does live PDF link checking yet. So, I created some PostScript-as-language code that can do link checking for you on a "semi-automated" basis.

Find this at *www.tinaja.com/psutils/graburls.ps*

The key secret is that links stay in plaintext, even when a PDF file is compressed and/or optimized. Thus, a typical Distiller-generated link might appear like so...

```
92 0 obj
<<
/A << /S /URI /URI (http://www.adobe.com)>>
/Type /Annot
/Subtype /Link
/Rect [ 38 362 73 374 ]
/C [ 0.7 0 0 ]
/Border [ 0 0 0 ]
>>
```

The assumption is made that each live link is preceeded by a /URI followed by exactly one space followed by an opening parenthesis character string. The PDF program is read one line at a time. All lines lacking this magic string are rejected. That part of your line beyond the magic string is searched to extract the url. This gets done by rejecting everything beyond the next occuring closing parenthesis. An adjustment is also made for extra long url's. These are identified by a trailing "\" on their first line.

The algorithm assumes that your PDF file has the magic prefix on the same line as your url and that there are no closing parenthesis inside the url itself. These assumptions may not be valid for all possible PDF code sources.

The extracted url is then reported to the Distiller log file and written to the output HTML file.

You change your source PDF and target HTML filenames in this routine with an editor or word processor. Then you send it to Distiller. A list of all links appears in the Distiller log file. You can visually inspect this file for obvious typos.

Separately, a new HTML document is created that lists only the links. You then upload this doc to your website and run Dr. HTML or another link checker.

Thus, you run GRABURLS.PS, next upload the generated GRABURLS.HTML file and then request Dr. HTML to check the extracted links for you.

No attempt is made to sort the url's alphabetically or eliminate dups. But Dr. HTML does this automatically.

A final possibility is to make a game out of it. Every now and then, offer something reasonably valuable in exchange for first reports of a dozen broken links or typos.

## For More Help

Examples of "HTML-like" menu buttons and boxes are at the end of this and my other .PDF files. Unlike HTML, the Acrobat menus and buttons are all in one file. With care, PDF files can be much shorter than HTML, load faster, and look significantly better.

Additional support on these and other PDF concepts is at my *www.tinaja.com/acrob01.html* and *www.tinaja.com/weblib01.html* Consulting help is at *www.tinaja.com/info01.html* and *www.tinaja.com/consul01.html*

Let's hear from you ✦

*Microcomputer pioneer and guru Don Lancaster is the author of 35 books and countless articles. Don maintains a US technical helpline you'll find at (520) 428-4073, besides offering all his own books, reprints and consulting services.*

*Don has a free new catalog crammed full of his latest insider secrets waiting for you. Your best calling times are 8-5 weekdays, Mountain Standard Time.*

*Don is also the webmaster of www.tinaja.com You can also reach Don at Synergetics, Box 809, Thatcher, AZ 85552. Or you can use email via don@tinaja.com*

**PLEASE CLICK HERE TO...**

- ▶ **Get a Synergetics catalog**
- ▶ **Start your tech venture**
- ▶ **Sponsor a display banner**
- ▶ **Find research solutions**
- ▶ **Send Don Lancaster email**
- ▶ **Pick up surplus bargains**
- ▶ **Find out what a tinaja is**
- ▶ **View recommended books**