PDF with Embedded Links

Author's note: I'd like to offer a big thank you to Christian Schmitz of Monkeybread Software, not only for creating and continually improving a fantastic plug-in, but also for promptly and patiently answering my many questions.

A few weeks ago we looked at creating a table of contents for a PDF containing nine individual reports. The article ended by asking a somewhat rhetorical question…

Now that we have a table of contents for our master report, wouldn't it be nice if we could hyperlink each TOC entry to its corresponding report within the PDF?

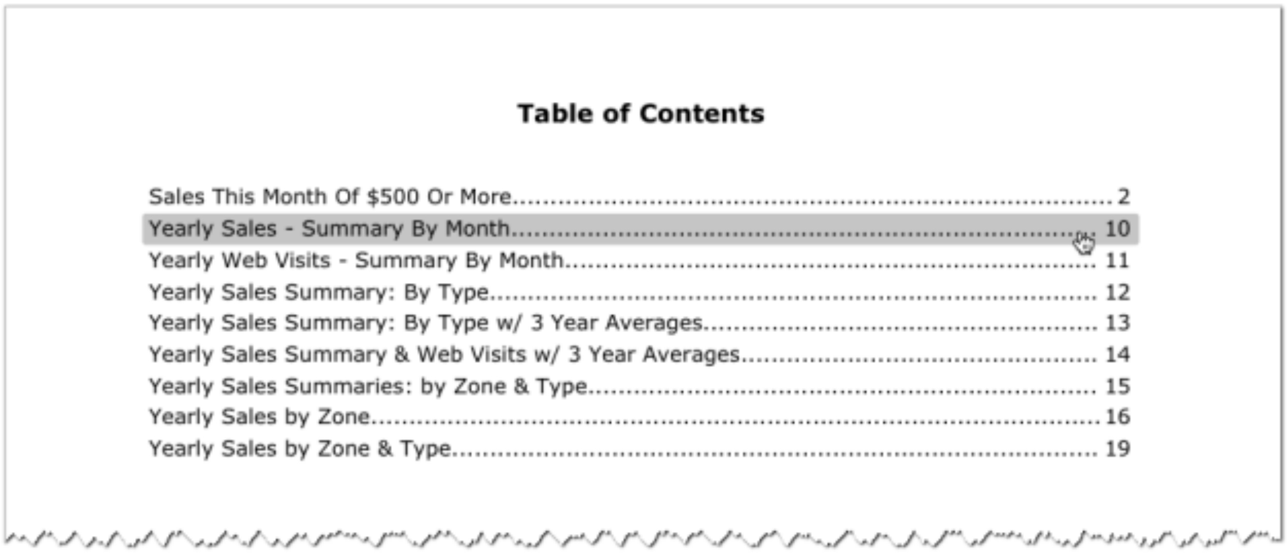


Table of Contents

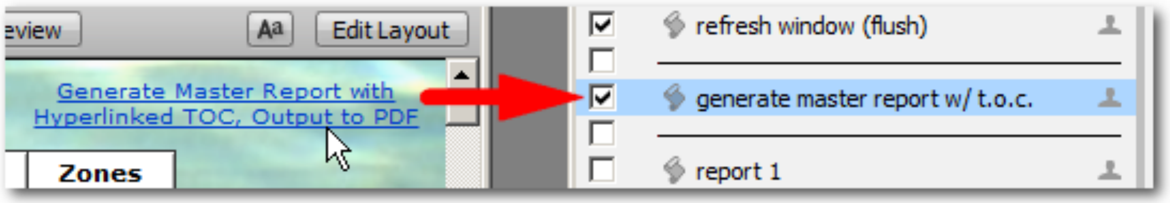
Sales This Month Of $500 Or More........ 2
Yearly Sales - Summary By Month........ 10
Yearly Web Visits - Summary By Month........ 11
Yearly Sales Summary: By Type........ 12
Yearly Sales Summary: By Type w/ 3 Year Averages........ 13
Yearly Sales Summary & Web Visits w/ 3 Year Averages........ 14
Yearly Sales Summaries: by Zone & Type........ 15
Yearly Sales by Zone........ 16
Yearly Sales by Zone & Type........ 19

FileMaker doesn't provide a built-in way to accomplish this, but it can be done using a combination of the MonkeyBread plug-in + DynaPDF Lite, and this will be explored in my next article.

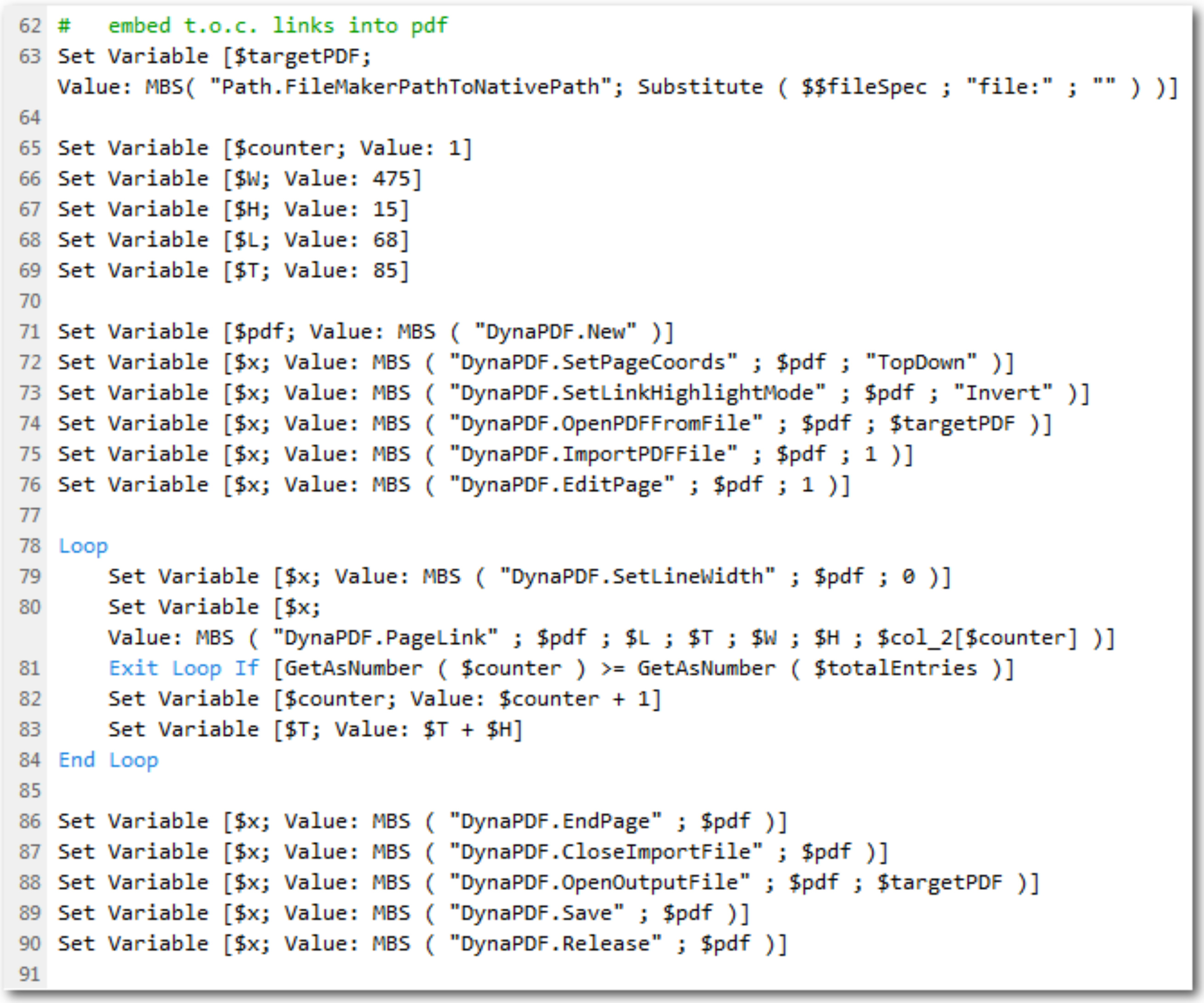
…and today we're going to dig into the nuts and bolts of what it takes to make this happen.

As usual, we have a demo file (Embedded PDF Links), but since it will only work for developers having licensed copies of the above-mentioned products, here is a PDF showing the end result: Report with Hyperlinked TOC.pdf

As before, we're going to concentrate on this particular script…


eview
Aa
Edit Layout
Generate Master Report with Hyperlinked TOC, Output to PDF
Zones
refresh window (flush)
generate master report w/ t.o.c.
report 1

…which hasn't changed since last time except for the addition of these steps:



```
#   embed t.o.c. links into pdf
Set Variable [$targetPDF;
Value: MBS( "Path.FileMakerPathToNativePath"; Substitute ( $$fileSpec ; "file:" ; "" ) )]

Set Variable [$counter; Value: 1]
Set Variable [$W; Value: 475]
Set Variable [$H; Value: 15]
Set Variable [$L; Value: 68]
Set Variable [$T; Value: 85]

Set Variable [$pdf; Value: MBS ( "DynaPDF.New" )]
Set Variable [$x; Value: MBS ( "DynaPDF.SetPageCoords" ; $pdf ; "TopDown" )]
Set Variable [$x; Value: MBS ( "DynaPDF.SetLinkHighlightMode" ; $pdf ; "Invert" )]
Set Variable [$x; Value: MBS ( "DynaPDF.OpenPDFFromFile" ; $pdf ; $targetPDF )]
Set Variable [$x; Value: MBS ( "DynaPDF.ImportPDFFile" ; $pdf ; 1 )]
Set Variable [$x; Value: MBS ( "DynaPDF.EditPage" ; $pdf ; 1 )]

Loop
    Set Variable [$x; Value: MBS ( "DynaPDF.SetLineWidth" ; $pdf ; 0 )]
    Set Variable [$x;
    Value: MBS ( "DynaPDF.PageLink" ; $pdf ; $L ; $T ; $W ; $H ; $col_2[$counter] )]
    Exit Loop If [GetAsNumber ( $counter ) >= GetAsNumber ( $totalEntries )]
    Set Variable [$counter; Value: $counter + 1]
    Set Variable [$T; Value: $T + $H]
End Loop

Set Variable [$x; Value: MBS ( "DynaPDF.EndPage" ; $pdf )]
Set Variable [$x; Value: MBS ( "DynaPDF.CloseImportFile" ; $pdf )]
Set Variable [$x; Value: MBS ( "DynaPDF.OpenOutputFile" ; $pdf ; $targetPDF )]
Set Variable [$x; Value: MBS ( "DynaPDF.Save" ; $pdf )]
Set Variable [$x; Value: MBS ( "DynaPDF.Release" ; $pdf )]
```

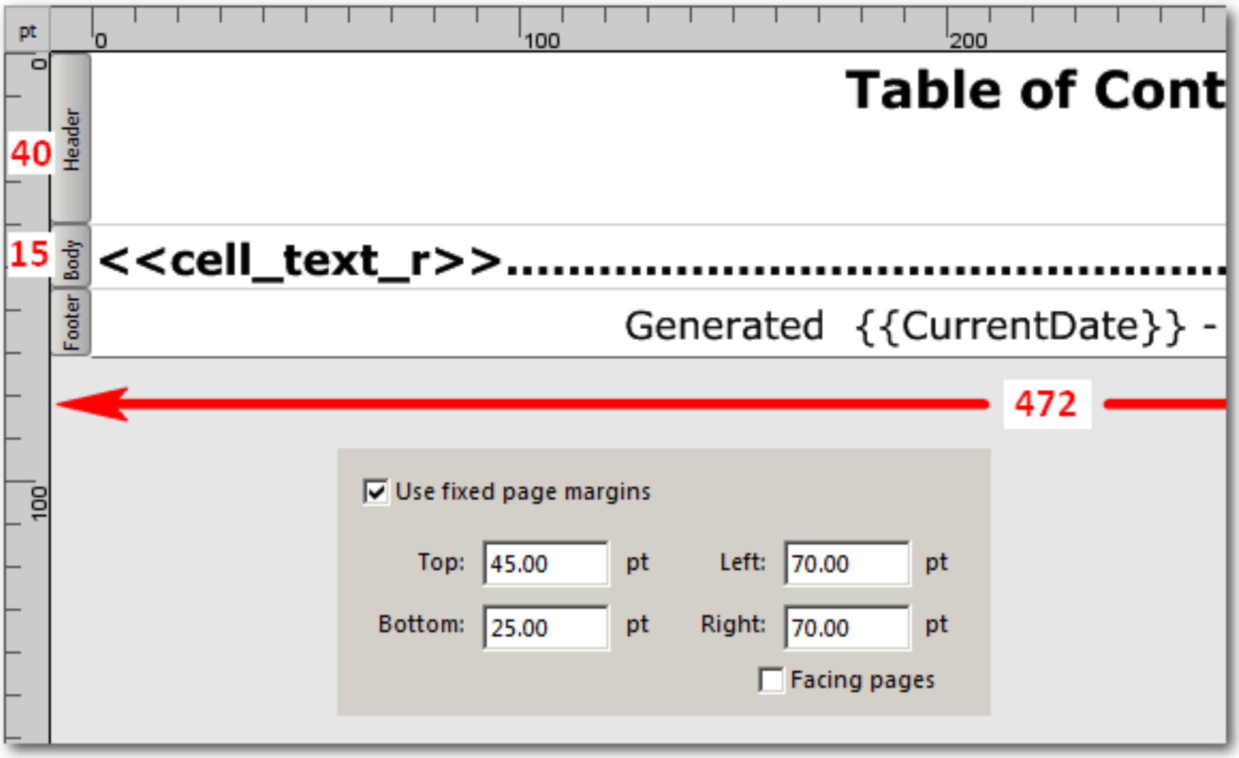
And now is probably a good time to mention that every MBS function has its own documentation page (see overview here) and the same holds true for DynaPDF functions.

We'll take a closer look some of these steps in just a minute. First, though, I want to say a little bit about PDF page linking… how it works in general, and the particular approach I'm taking here.

A PDF page link is created by specifying a set of rectangular coordinates, and, in effect, "drawing" an invisible rectangle at a certain location on a given page of a PDF, and then specifying the target page to go to (within that same PDF) when that link is clicked. In today's article and demo this is accomplished via a function called "DynaPDF.PageLink".

As chance would have it, a table of contents is ideally suited to this approach, because the entries are predictably located on the T.O.C. page, making it easy to programmatically determine where to place the invisible page link rectangles.

The layout dimensions can help us determine the width and height of the rectangle, as well as the starting position (top, left).

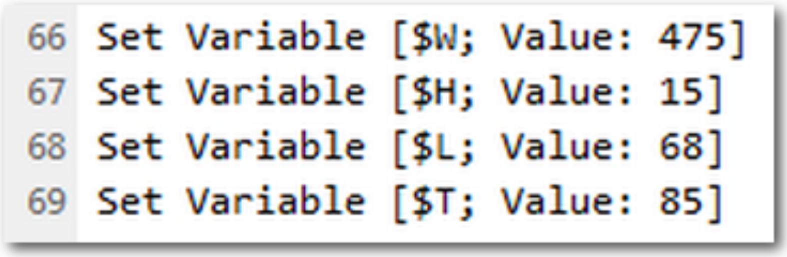

pt
0
100
200
Table of Cont
40
Header
15
Body
<<cell_text_r>>..
Footer
Generated {{CurrentDate}} -
472
100

- [✓] Use fixed page margins

Top:	45.00	pt	Left:	70.00	pt
Bottom:	25.00	pt	Right:	70.00	pt

- [ ] Facing pages

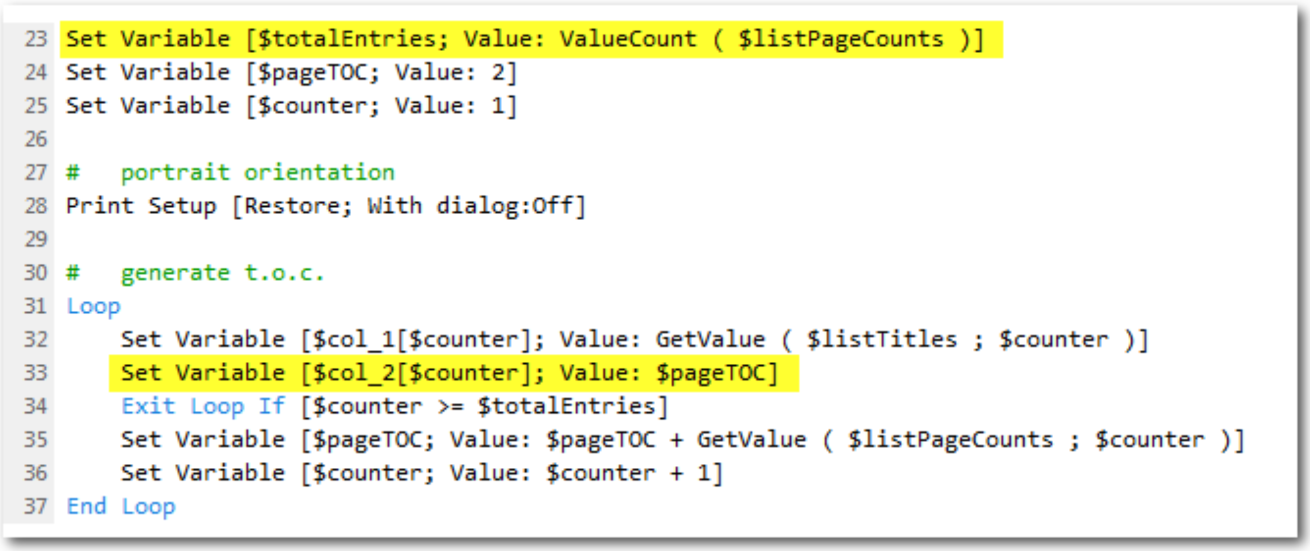
Based on the above we're going to use these coordinates for width, height, left and top.



```
Set Variable [$W; Value: 475]
Set Variable [$H; Value: 15]
Set Variable [$L; Value: 68]
Set Variable [$T; Value: 85]
```

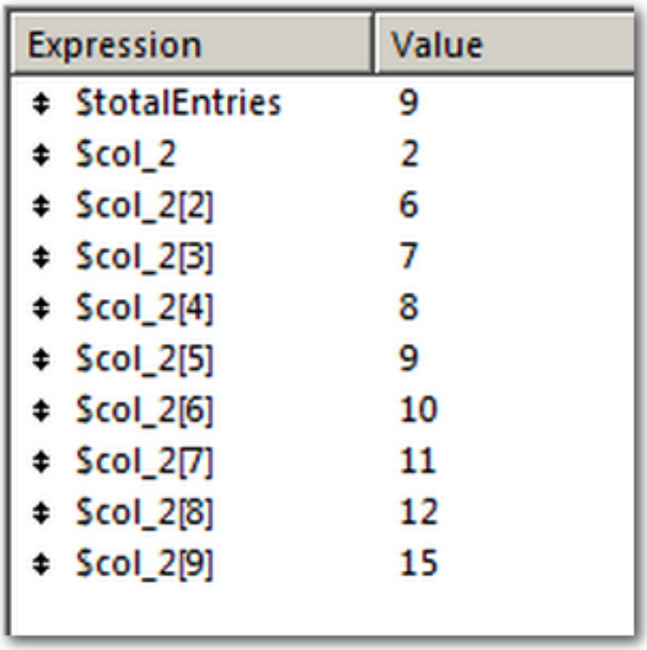
The first three will remain constant, but $T (top) will be incremented by $H (the rectangle height, i.e., 15) for each TOC entry, as the script works its way down the page drawing invisible page link rectangles.

As mentioned last time, the PDF will contain either eight or nine reports, so the TOC will have either eight or nine entries (and corresponding page values, which we will use as the target for our page links)… and, conveniently, our script has already loaded these values into variables, as part of the basic TOC generation process, as per these two highlighted steps.



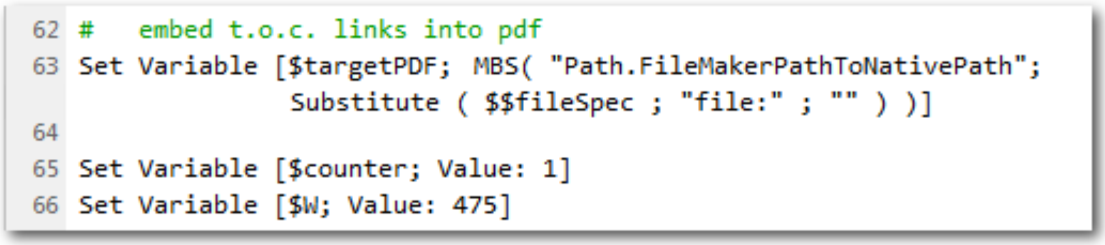
```
Set Variable [$totalEntries; Value: ValueCount ( $listPageCounts )]
Set Variable [$pageTOC; Value: 2]
Set Variable [$counter; Value: 1]

#    portrait orientation
Print Setup [Restore; With dialog:Off]

#    generate t.o.c.
Loop
    Set Variable [$col_1[$counter]; Value: GetValue ( $listTitles ; $counter )]
    Set Variable [$col_2[$counter]; Value: $pageTOC]
    Exit Loop If [$counter >= $totalEntries]
    Set Variable [$pageTOC; Value: $pageTOC + GetValue ( $listPageCounts ; $counter )]
    Set Variable [$counter; Value: $counter + 1]
End Loop
```

Once the above portion of the script has completed, $totalEntries and the repeating variable, $col_2, will contain something like this.



Expression	Value
$totalEntries	9
$col_2	2
$col_2[2]	6
$col_2[3]	7
$col_2[4]	8
$col_2[5]	9
$col_2[6]	10
$col_2[7]	11
$col_2[8]	12
$col_2[9]	15

Okay, let’s take a closer look at some of the page link script steps.



```
62 #   embed t.o.c. links into pdf
63 Set Variable [$targetPDF; MBS( "Path.FileMakerPathToNativePath";
                Substitute ( $$fileSpec ; "file:" ; "" ) )]
64
65 Set Variable [$counter; Value: 1]
66 Set Variable [$W; Value: 475]
```

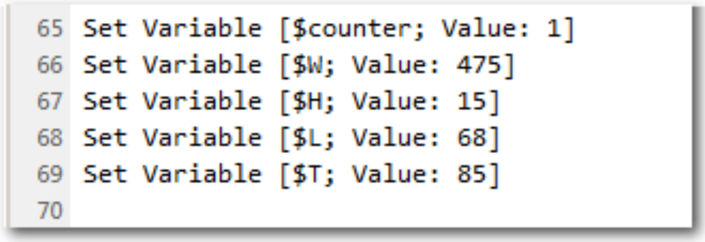
Step 63 transforms the "Save As PDF"-friendly path + filename in $$fileSpec, e.g.,

```
file:/C:/Users/Kevin/Desktop/report.pdf   (Windows)
file:/MacHD/Users/Kevin/Desktop/report.pdf  (Macintosh)
```

...into one that DynaPDF can use, e.g.,

```
C:\Users\Kevin\Desktop\report.pdf   (Windows)
/Users/Kevin/Desktop/report.pdf   (Macintosh)
```

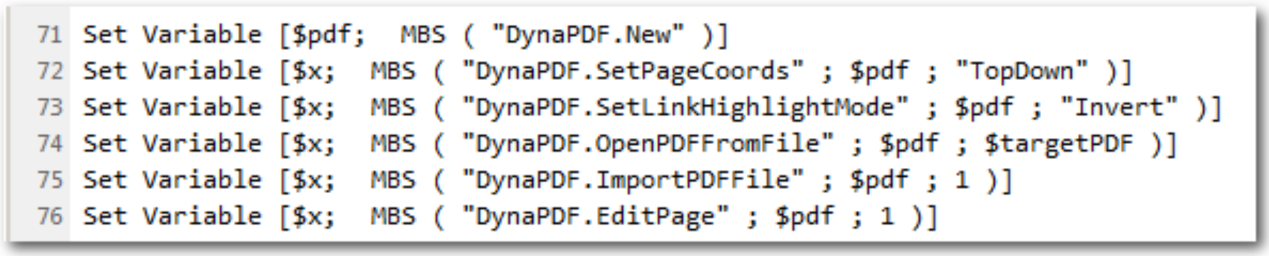
The $counter variable initialized in step 65 will be used in step 80 to help extract page number values from the $col_2[repetitions] discussed above.



```
65 Set Variable [$counter; Value: 1]
66 Set Variable [$W; Value: 475]
67 Set Variable [$H; Value: 15]
68 Set Variable [$L; Value: 68]
69 Set Variable [$T; Value: 85]
70
```

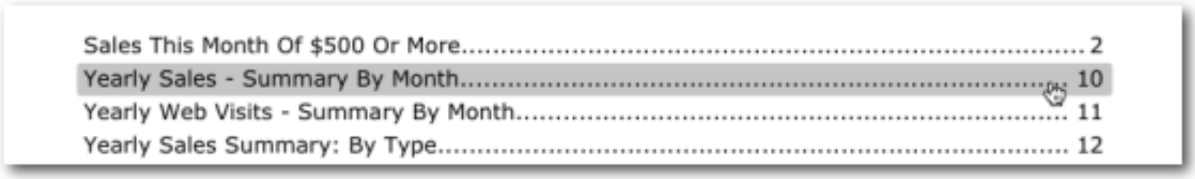
Step 71 begins the DynaPDF process, and as per the "DynaPDF.New" help page, there should only be one instance of this running in memory at a time (see step 90 note below).

Step 72 tells DynaPDF to begin coordinates at the top left of the page (instead of using the native PDF coordinate system, which starts at the bottom left)...



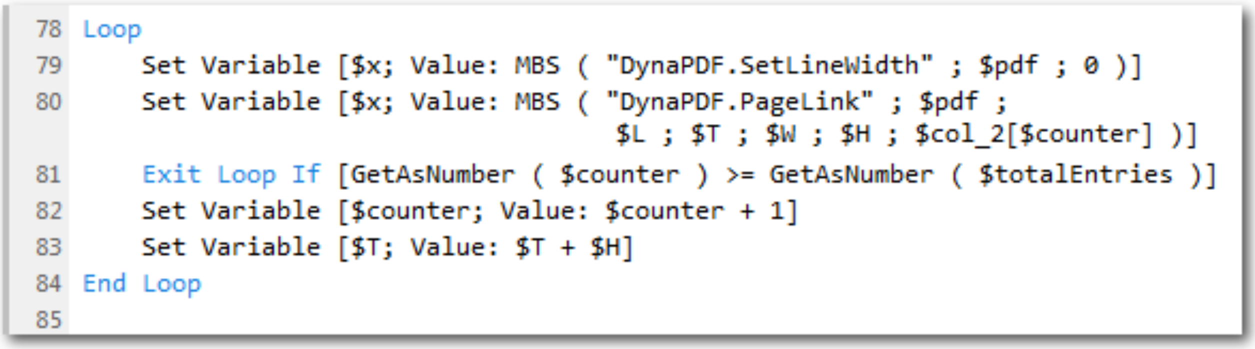
```
71 Set Variable [$pdf;  MBS ( "DynaPDF.New" )]
72 Set Variable [$x;  MBS ( "DynaPDF.SetPageCoords" ; $pdf ; "TopDown" )]
73 Set Variable [$x;  MBS ( "DynaPDF.SetLinkHighlightMode" ; $pdf ; "Invert" )]
74 Set Variable [$x;  MBS ( "DynaPDF.OpenPDFFromFile" ; $pdf ; $targetPDF )]
75 Set Variable [$x;  MBS ( "DynaPDF.ImportPDFFile" ; $pdf ; 1 )]
76 Set Variable [$x;  MBS ( "DynaPDF.EditPage" ; $pdf ; 1 )]
```

…and step 73 determines what the user will see when they click on a PDF link (roughly equivalent to the button “pressed” state in FileMaker), e.g.,


Sales This Month Of $500 Or More... 2
Yearly Sales - Summary By Month... 10
Yearly Web Visits - Summary By Month.. 11
Yearly Sales Summary: By Type.. 12

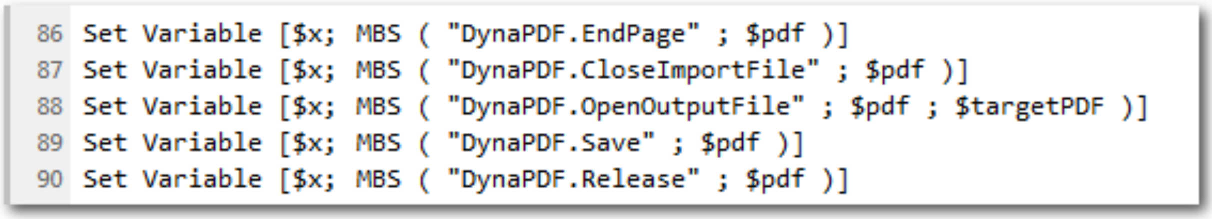
Step 79 ensures that the rectangle is invisible, i.e., does not display a border, if the user opens the PDF in Acrobat or Acrobat Reader. (The rectangle is always invisible in Preview.)

Step 80 draws the page link rectangles and assigns the target page numbers.


```
78 Loop
79     Set Variable [$x; Value: MBS ( "DynaPDF.SetLineWidth" ; $pdf ; 0 )]
80     Set Variable [$x; Value: MBS ( "DynaPDF.PageLink" ; $pdf ;
                                  $L ; $T ; $W ; $H ; $col_2[$counter] )]
81     Exit Loop If [GetAsNumber ( $counter ) >= GetAsNumber ( $totalEntries )]
82     Set Variable [$counter; Value: $counter + 1]
83     Set Variable [$T; Value: $T + $H]
84 End Loop
85
```

Step 87 is necessary on the Windows OS, but not on the Mac.

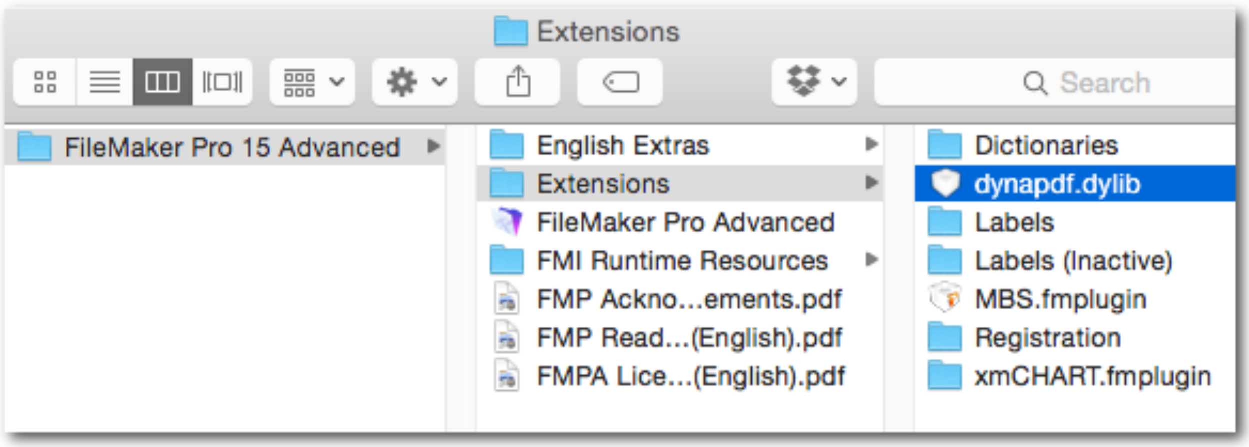
Step 90 frees up memory claimed by “DynaPDF.New” in step 71, and is not optional.


```
86 Set Variable [$x; MBS ( "DynaPDF.EndPage" ; $pdf )]
87 Set Variable [$x; MBS ( "DynaPDF.CloseImportFile" ; $pdf )]
88 Set Variable [$x; MBS ( "DynaPDF.OpenOutputFile" ; $pdf ; $targetPDF )]
89 Set Variable [$x; MBS ( "DynaPDF.Save" ; $pdf )]
90 Set Variable [$x; MBS ( "DynaPDF.Release" ; $pdf )]
```

And now you know everything I know (as of Feb 2017, at any rate) about generating embedded PDF page links.

Postscript: MBS + DynaPDF Installation and Registration

On the Macintosh, I’ve chosen to install the DynaPDF library file in the FileMaker Extensions folder, along with the MBS plug-in.

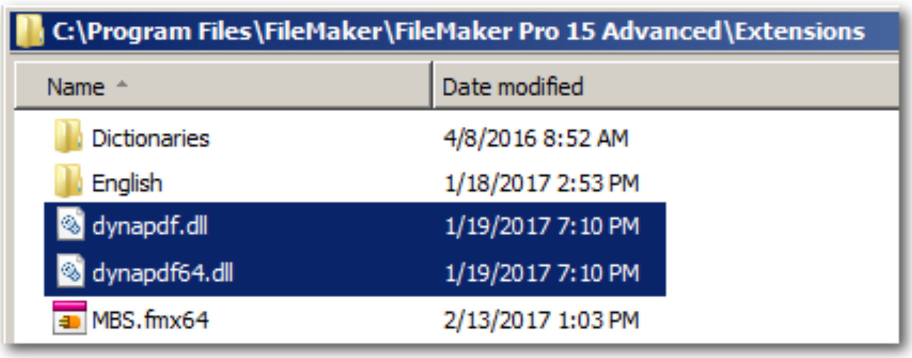


FileMaker Pro 15 Advanced

English Extras
Extensions
FileMaker Pro Advanced
FMI Runtime Resources
FMP Ackno...ements.pdf
FMP Read...(English).pdf
FMPA Lice...(English).pdf

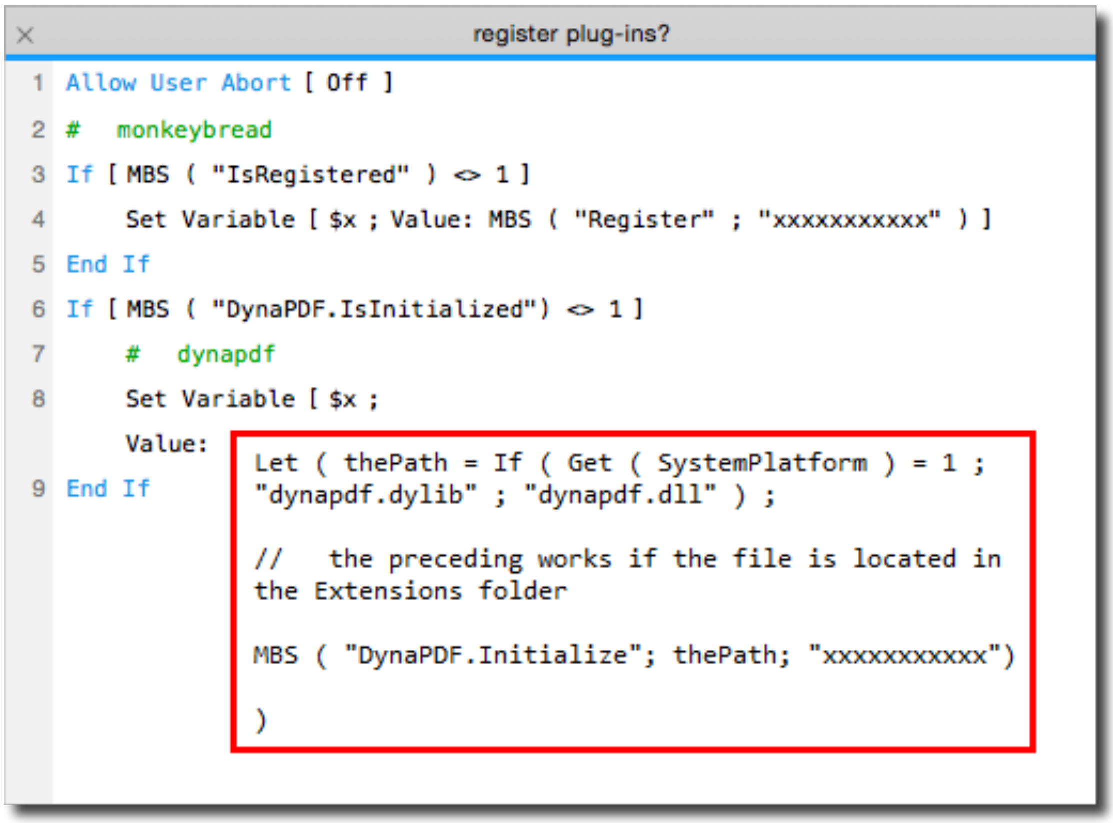
Dictionaries
dynapdf.dylib
Labels
Labels (Inactive)
MBS.fmplugin
Registration
xmCHART.fmplugin

On Windows, DynaPDF consists of these two files, and again, I've chosen to install them in the FileMaker Extensions folder along with the MBS plug-in.


C:\Program Files\FileMaker\FileMaker Pro 15 Advanced\Extensions

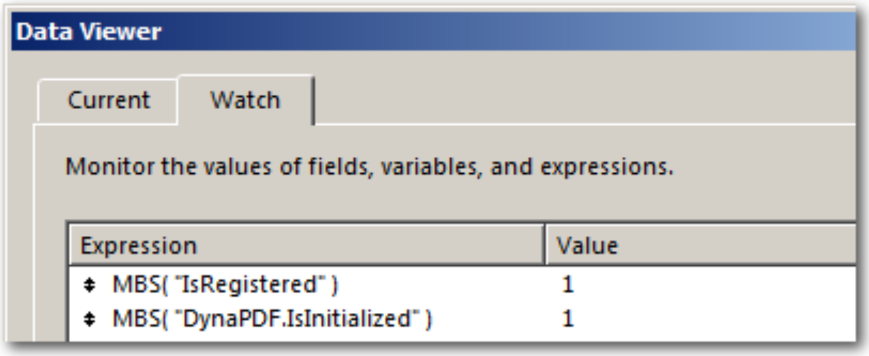
Name	Date modified
Dictionaries	4/8/2016 8:52 AM
English	1/18/2017 2:53 PM
dynapdf.dll	1/19/2017 7:10 PM
dynapdf64.dll	1/19/2017 7:10 PM
MBS.fmx64	2/13/2017 1:03 PM

The startup script in the demo file calls this script (obviously, you will need to substitute the xxxxxxxxxx-es with your registration info — and note that there are four versions of DynaPDF, and the methodology we're looking at today requires at least the "Lite" version).



```
Allow User Abort [ Off ]
#  monkeybread
If [ MBS ( "IsRegistered" ) <> 1 ]
    Set Variable [ $x ; Value: MBS ( "Register" ; "xxxxxxxxxxx" ) ]
End If
If [ MBS ( "DynaPDF.IsInitialized") <> 1 ]
    #  dynapdf
    Set Variable [ $x ;
    Value: Let ( thePath = If ( Get ( SystemPlatform ) = 1 ;
           "dynapdf.dylib" ; "dynapdf.dll" ) ;

           //  the preceding works if the file is located in
           the Extensions folder

           MBS ( "DynaPDF.Initialize"; thePath; "xxxxxxxxxxx")

           )
End If
```

And, after running the "register" script, the Data Viewer confirms that all is well.



Monitor the values of fields, variables, and expressions.

Expression	Value
MBS("IsRegistered")	1
MBS("DynaPDF.IsInitialized")	1

This entry was posted in Level: Intermediate, Version: FM 13 or later and tagged PDF on February 21, 2017 [https://filemakerhacks.com/2017/02/21/pdf-with-embedded-links/] by Kevin Frank.