

Volume 4 · Number 4

X-RAY

MAGAZINE

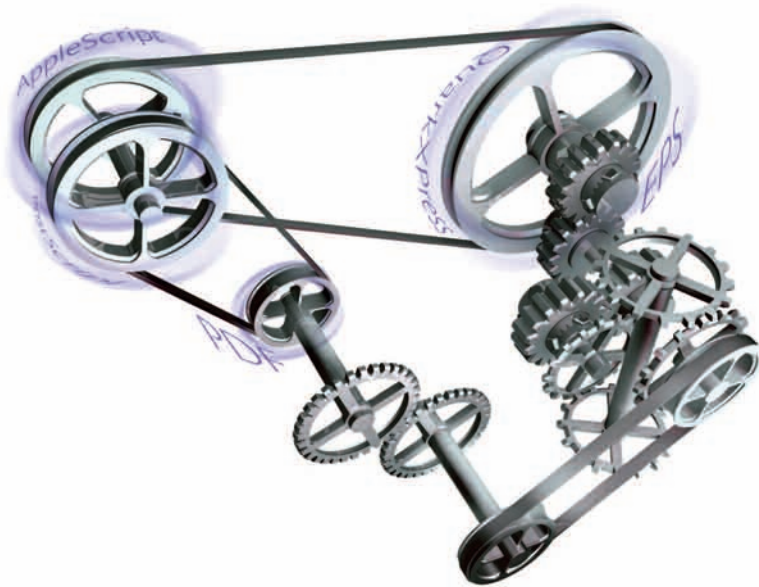
QUARK SOFTWARE WORKFLOW SOLUTIONS AND IMPLEMENTATION



Automating PDF Creation with AppleScript

BY BENJAMIN S. WALDIE

I am sure that QuarkXPress users can attest to the fact that there are many time-consuming and repetitive tasks that take place in any desktop-publishing workflow. I have been developing custom AppleScripts to automate just these types of tasks for my clients for years. In the many desktop-publishing automation scripts that I have developed, similarities can't help but be noticed, even in the most obscure workflows. In QuarkXPress-based workflows, one task that stands-out as a very commonly automated task is the process of using AppleScript to convert QuarkXPress documents to PDF.



As you probably know, this conversion process is not too difficult when performed manually. It is possible to manually export a QuarkXPress document to PDF by selecting the **FILE > EXPORT > LAYOUT AS PDF** menu. Unfortunately, the ability to perform this same function from AppleScript is not possible through direct scripting; nor is it possible to write an AppleScript that will print a document directly to PDF, at least not without the use of a virtual PDF printer of some type. These features are simply not AppleScript-savvy at this time. In situations such as these, sometimes there are workarounds that can be used to get around the limitations.

An application's non-scriptable features can often be automated by using a third-party tool to interact with the graphical user interface (GUI). In Mac OS X 10.3 and higher, Apple provides one mechanism, called GUI scripting (<http://www.apple.com/applescript/uisc scripting/>) that will allow you to do just this. Using GUI scripting, AppleScripts can be written to interact with an application in the same way that the user would interact with an application using the mouse and keyboard: by selecting menus, clicking buttons, typing text, and more. Unfortunately, GUI scripting does not always work with every application or with every non-scriptable feature of every application. In these situations, combining GUI scripting and an AppleScript-savvy macro program such as QuicKeys (<http://www.quickeys.com>) will usually allow you to work around such issues. Unfortunately, these workarounds are not always possible and when they are, they can often seem clumsy and unreliable.

There are two primary methods that are typically used by developers to convert QuarkXPress documents to PDF with AppleScript. One option is to write a script that will print the document to PostScript format, and then convert the resulting PostScript file to PDF. Another option is to write a script that will save the pages of the document as individual EPS, and then convert these files to PDF. We will discuss these options in detail throughout this column.

Please note that the example code specified in this document was written and tested with QuarkXPress version 6.5. Minor changes to Quark's AppleScript terminology sometimes occur between application versions. Therefore, some adjustments may be necessary to the code within this article in order for it to function with versions of QuarkXPress other than version 6.5.

Printing a PostScript File

When printing to PostScript format, you will probably want to adjust the print settings for your document. Fortunately, almost all of QuarkXPress' print settings are accessible and are modifiable directly with AppleScript. These print settings are broken up into several groups of settings, which are described below.

GENERAL PRINT SETTINGS

General print settings such as the printer type, paper size, orientation, and more, can be found in QuarkXPress' print setup record class, which is accessible through the print setup property of a document. The following example code demonstrates how to modify certain general print settings, in this case, setting the orientation to landscape and setting tiling to automatic.

```
tell application "QuarkXPress"
    tell document 1
        tell print setup
            set orientation to
            landscape
            set tiling to automatic
        end tell
    end tell
end tell
```

Again, this is an example for setting only two general print settings. For a complete list of modifiable general print settings refer to the print setup record class, located in the QuarkXPress Ancillary Objects suite, in QuarkXPress' AppleScript dictionary.

LAYER PRINT SETTINGS

To disable printing of specific layers in a document, you must adjust the suppress print property for each of the desired layers. The following example code demonstrates how this may be done. This code would disable printing of a layer named Production Notes.

```
tell application "QuarkXPress"
    tell document 1
        tell layer
            "Production Notes"
            set suppress print
            to true
        end tell
    end tell
end tell
```

BLEED PRINT SETTINGS

Bleed settings are controlled by adjusting properties of the custom bleeds setup class of the document. The following example code demonstrates how this is done.

```
tell application "QuarkXPress"
    tell document 1
        tell custom bleeds setup 1
            set bleed type to
            asymmetric
            set bleed to {"1\\"",
            "1\\"", "1\\"", "1\\""}
        end tell
    end tell
end tell
```

For a complete list of modifiable bleed print settings refer to the custom bleeds setup class, located in the Custom Bleeds Suite in QuarkXPress' AppleScript dictionary.

Almost all of QuarkXPress' print settings are accessible and are modifiable directly with AppleScript.

OPI PRINT SETTINGS

To modify the OPI print settings you may adjust the properties of the OPI setup class, which can be found in the OPI Setup Record suite in QuarkXPress' AppleScript dictionary. The following example code demonstrates how to make adjustments to these settings.

```
tell application "QuarkXPress"
    tell document 1
        tell OPI setup 1
            set OPI active to true
            set include TIFF to true
        end tell
    end tell
end tell
```

For a complete list of modifiable OPI print properties, refer to the OPI setup class in QuarkXPress' AppleScript dictionary.

FONT INCLUSION

In Mac OS 9, one primary area of concern when printing documents to PostScript format with AppleScript was font inclusion. Unfortunately, the ability to embed fonts in the resulting PostScript file was not an AppleScript-accessible setting. Therefore, in order to embed fonts, it was necessary to utilize a third-party AppleScript tool, such as Prefab Player (<http://www.prefab.com>) in order to navigate the print dialogs and choose the appropriate font inclusion options. With Mac OS X, font inclusion is no longer an issue. When printing a document to PostScript format, Mac OS X will automatically embed fonts for you.

PRINTING TO POSTSCRIPT FORMAT

Once your script has made any necessary adjustments to any print settings, it is ready to print the document to PostScript format. This action is performed by using the print command and specifying an output path for the command's PostScript file parameter. The following example code demonstrates how this is done.

```
set theOutputFolder to
(chOOSE folder) as string

set thePostScriptFilePath to
theOutputFolder & "output.ps"

tell application "QuarkXPress"
    tell document 1
        print PostScript file thePostScriptFilePath
    end tell
end tell
```

NOTE: If you are using Mac OS X 10.4, please be aware that changes were made to the operating system's print architecture that can result in QuarkXPress 6.5 crashing when printing with AppleScript. To resolve this problem, download and install the Output Enhancements XTensions software for QuarkXPress, which can be found via the **SUPPORT > DESKTOP DOWNLOADS** section of Quark's web site, <http://www.quark.com/service/desktop/downloads/details.jsp?idx=601>

Saving an EPS File

Saving a document's pages in EPS format is an alternative to printing a document to PostScript format. Perhaps the most noticeable difference though, is that you cannot embed fonts in the resulting EPS files.

EPS BLEED SETTINGS

When saving a document's pages in EPS format, it is possible to specify the bleed settings to be used. Like printing to PostScript, these settings may be modified by adjusting properties of the custom bleeds setup class. This class possesses specific properties for modifying the bleed for EPS files, as demonstrated in the following example code.

```
tell application "QuarkXPress"
    tell document 1
        tell custom bleeds setup 1
            set EPS bleed type to
            asymmetric
            set EPS bleed to {"1\\"",
            "1\\"", "1\\"", "1\\""}
        end tell
    end tell
end tell
```

Again, for a complete list of custom bleed settings, consult QuarkXPress' AppleScript dictionary.

SAVING A PAGE AS AN EPS

Saving a document page in EPS format is fairly straightforward. This is done with the save command, as demonstrated by the code below. This code will prompt the user to select an output folder. It will then save page 1 of the current QuarkXPress document as an EPS into the specified output folder, naming the file output.eps.

```
set theOutputFolder to
(chOOSE folder) as string

set theEPSFilePath to
theOutputFolder & "output.eps"

tell application "QuarkXPress"
    tell document 1
        save page 1 in file
        theEPSFilePath
    end tell
end tell
```

OPTIONAL SAVE SETTINGS

When saving a page of a document in EPS format there are a number of optional parameters that may be specified, if desired. These settings include the EPS format, OPI settings, scaling, and more. The following example code demonstrates how some of these optional settings may be used in conjunction with the save command to save a document page in EPS format.

```

set theOutputFolder to
(choose folder) as string
set theEPSFilePath to
theOutputFolder & "output.eps"
tell application "QuarkXPress"
  tell document 1
    save page 1 in file
    theEPSFilePath EPS format
    Mac color OPI include images
  end tell
end tell

```

PDF Conversion Options

Once your QuarkXPress document has been output in either PostScript or eps format, it is ready to be converted to pdf format. There are multiple ways that this can be done.

One way is to implement the Acrobat Distiller software (<http://www.adobe.com/products/acrobat/>). Acrobat Distiller gives you the ability to pre-configure settings to be utilized when converting files to PDF format. Distiller is also AppleScript-savvy, making it possible for a script to be written to pass files directly to Distiller for conversion to PDF. For information about scripting Distiller directly, consult its AppleScript dictionary. As an alternative to direct scripting, Distiller also includes the ability to configure watched folders, where any detected acceptable files will be automatically converted to PDF format and saved into an output folder. Using Distiller's watched-folder capabilities, an AppleScript can be written to place PostScript or EPS files into an input folder being watched by Distiller, and then monitor the corresponding output folder for the converted PDF files to appear. Once detected, the script can resume processing the PDF files.

Another way that PostScript and eps files can be converted to pdf format is through the use of a utility that is built into the Mac os x printing architecture. This utility must be triggered from unix, using the do shell script AppleScript command in Mac os x, as demonstrated by the example code below. This code will prompt the user to select an eps or PostScript file, as well as an output folder. It will then convert the specified file to pdf, and save it into the specified output folder as a file named output.pdf.

```

set theFileToConvert to (choose
file with prompt "Select an EPS or
a PostScript file:") as string
set theOutputFolder to (choose
folder) as string
set theFileToConvert to quoted
form of (POSIX path of theFileTo-
Convert as string)
set thePDFFilePath to quoted form
of (POSIX path of (theOutputFolder
& "output.pdf") as string)
do shell script
"/System/Library/Printers/Li-
braries/./convert -f " & theFile-
ToConvert & " -o " &
thePDFFilePath & " -j \"applica-
tion/pdf\" "

```

Using this method of PDF generation you are unfortunately not able to specify various output settings, as you can when utilizing Distiller. Therefore, the resulting PDF file will be equivalent to that of a PDF that is printed using Apple's Save as PDF option, which is accessible when manually printing a file from within any application. See figure 1.

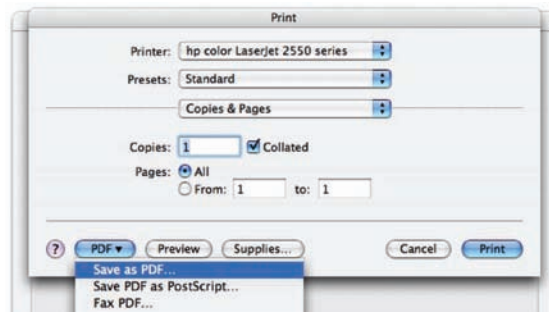


fig 1 ▲ Printing a Document to PDF Manually

In Conclusion

Hopefully, the information in this column will shed some light on how many QuarkXPress users are automating the conversion of their documents to PDF format using AppleScript. There are some other techniques floating around here and there, however, the methods that I have discussed seem to be the most commonly utilized.

If you are interested in automating the process of converting your QuarkXPress documents to PDF format with AppleScript, then the methods suggested in this column should provide you with a good place to start. I would encourage you to further explore these methods to see how they might fit into your own unique workflow.

See you in the trenches. ☒