T E C H N O T E:
Mixing QuickDraw & PostScript Printing from Your App: Some Gotchas

By Dave Polaschek
dpolasch@apple.com
Apple Developer Technical Support (DTS)

If your application generates PostScript directly, or you're considering writing an app that does, there are some techniques that will help your development efforts. This Technote explains these techniques, as well as gotchas that are either inadequately documented, or scattered across multiple documentation sources.

This Note is aimed at application developers who want to mix the PostScript and QuickDraw imaging models.

Using the PostScriptHandle PicComment, a Friend Indeed

Think of the PostScriptHandle `PicComment` as your friend. Use it, but be aware of its limitations. Because this `PicComment` lets you send huge chunks of PostScript directly to a printer, it's your obvious choice. However, there are a few gotchas that you need to consider.

Gotcha #1

The LaserWriter driver has control of your coordinate system.

This means that the normal PostScript coordinate system you know and love has been transformed, i.e., coordinates now match the QuickDraw orientation. This actually makes things easier for you. When you include the QuickDraw representation of the PostScript you're sending, the coordinate system will be the same as the one you used for the PostScript. You don't have to think about converting the coordinates. If you *do* include a QuickDraw PICT so you can print to QuickDraw printers, you should bracket this QuickDraw representation with the `PostScriptBegin` and `PostScriptEnd PicComments`.

Also, if the user selects 2-up or a similar option in the print driver, all the coordinates get transformed behind your back by the driver. The consequence of your app "thinking" it knows the coordinate system is that your app may end up coloring outside the lines. That's another good reason to keep things simple so you don't outsmart yourself.

Gotcha #2

The QuickDraw side of the imaging engine in the print driver isn't aware of any changes you make to the PostScript imaging state.

The LaserWriter driver attempts to save and restore the graphics state around Postscript that you're sending to the printer, but there are cases when you can confuse it. In one case, for example, your app generates PostScript which prints to multiple pages without telling the driver. Another example would be changing the PostScript pen size directly rather than via QuickDraw. In general, if you are changing the state, you need to restore it.

Gotcha #3

QuickDraw, and the Printing manager only know about pages when you signal page boundaries by calling `PrOpenPage` *and* `PrClosePage`.

What this means to you is that any PostScript you send using the `PostScriptHandle PicComment` should be a single image. More specifically, it should meet the criteria established for an EPS image, as described in Adobe's *PostScript Language Reference Manual, 2nd Edition, Appendix H.* Following these guidelines will also help you avoid Gotcha #2.

Printing PostScript Files

If you want to print complete PostScript files to a PostScript printer, you should be using the `PAP.WrkStation.o` library, which is provided on the Mac OS SDK CD. Sample code for using that library is in the works, and will appear on a future Developer CD.

Note
Be sure that you're using the latest version of the library in order to avoid compatibility problems and to get the latest bug fixes from Apple. ◆

Summary

The `PostScriptHandle PicComment` is your friend, but even as your friend, you need to be aware of its limitations.

Further Reference

- *Inside Macintosh: Imaging with QuickDraw,* Appendix B
- *PostScript Language Reference Manual, 2nd Edition*, Appendix H

Acknowledgments

Thanks to Ben Edlund for creating “The Tick” and John Warnock, et al., for creating PostScript.

Thanks to Ingrid Kelly, Dave Hersey, and Guillermo Ortiz.