LaserWriter Techniques

How to optimize print code for the LaserWriter

Although the Print Manager was originally designed to allow applications to print in a device-independent manner, there are things about the LaserWriter that have to be addressed in a printer-dependent way.

With the addition of picComments and the **PrGeneral** procedure, your application should never need to know what type of device it is connected to. However, some developers feel that their applications should be able to take advantage of all the features provided by a particular device, not just those of the Print Manager. Apple does not recommend writing device dependent code, but Apple Technical Notes #72 and #91 describe some ways of optimizing LaserWriter code. A summary of those Technical Notes follows. Unsupported procedures from these Technical Notes will not be covered.

When printing to a LaserWriter, all Quickdraw calls are translated (via Quickdraw bottlenecks) into PostScript, which is in the LaserWriter ROM. Most Quickdraw operations are supported in PostScript. The LaserWriter driver does not support:

- XOR and NotXOR transfer modes.
- The grafverb invert.
- SetOrigin calls within PrOpenPage and PrClosePage calls for driver versions prior to 3.0 (use OffsetRect instead). Driver versions starting with 3.0 do not have this problem.
- Regions are ignored. You can simulate regions using polygons or bitmaps. See Creating Offscreen Bitmaps for more information.
- Clip regions should be limited to rectangles.
- Standard graphical objects, like rectangles, circles, etc., will be the same size on the LaserWriter as they appear on the screen. There are, however, two types of objects where this is not the case: text and bitmaps.

Because of the different resolution capabilities, there is a difference between character widths on the screen and on the printed page. However, to make sure your lines still break in the same place, the driver changes the word spacing and character spacing to keep line endpoints where you want them. What this means is that you can't count on the positions or the widths of printed characters being exactly the same as they are on the screen.

If you are doing your own line layout, i.e., positioning the words on the line yourself, you may want to disable the LaserWriter's line layout routines. 128K ROMs and later support the specification of fractional pixel widths for screen fonts, increasing the screen-to-printer accuracy. This fractional-width feature is disabled by default. To enable it, you can use the **SetFractEnable** procedure, after you call **InitFonts**. You can use picComments to left-; right-; or center-justify text. Only the left, right, or center endpoints will be accurate. If the text is fully justified, both endpoints will be accurate.

Apple has published several technical notes that strongly recommend you use device-indpendent printing techniques. Even though some devices can physically support non-standard page sizes, Apple recommends using only those page sizes directly supported by the printer driver. Specifically, they say not to change the page sizes in the "Style" dialog. Also, to be compatible with future "printer-like" devices, you have to make sure that the application prints in a device-independent manner. Avoid testing undocumented fields, setting fields in the print record directly, and bypassing the existing print dialogs.

The following techniques may help your application print faster in a device-independent manner

- Wherever possible, avoid the Quickdraw Erase calls (i.e. **EraseRect**, EraseOval, etc.). It takes a lot of time to handle the erase function because every bit (90,000 bits/ sq. in.) has to be cleared. Erasing is unnecessary because the paper does not need to be erased the way the screen does.
- A good deal of the time involved in printing patterns goes into building the pattern's bitmap. PostScript-optimized patterns, such as black, white, and all the gray patterns will produce output much faster than non-optimized patterns. Using a different pattern works, it just takes longer.
- Try to avoid changing fonts frequently. Characters are built and cached (if there is enough room) as they are needed. When the font is changed, the character has to be built again. Also, if the font is not in the LaserWriter, it has to be downloaded from the Macintosh. Using font substitution might be a simple solution to this problem.
- Avoid using **TextBox**. It makes calls to **EraseRect** for every line of text it draws and all of those calls to **EraseRect** slow the printer down. You might want to use a different method of displaying text (like **DrawString** or **DrawText**) or write your own version of **TextBox**. If you are making **TextBox** calls now, switching to something else can improve your speed on the order of 5 to 1.
- Because of the way rectangle intersections are determined, a clip region that falls outside of the rPage rectangle, will substantially slow the printer. By making sure your clip region is entirely within the rPage rectangle, you can improve your printing by approximately 400 percent.
- Do not use spool-a-page/print-a-page as some applications do when printing on the ImageWriter.
- Using DrawChar to place every character you want to print can take a long time. That's because there are bottlenecks for each character you draw. Also, the printer driver spends a lot of time in trying its best to do line layout -- making sure the character spacing is just right. In the 3.0 drivers and later, there are picture comments that turn off the line layout optimization and alleviate some of the problem.
- Avoid using empty Quickdraw objects. Quickdraw objects that are empty (meaning they have no pixels in them) and are filled but not framed, will not print on the ImageWriter and will not show up on the screen. On the LaserWriter they are real objects and will be printed.

When clipping characters out of a string, make sure that the clipping rectangle or region is greater than the bounding box of the text you want to clip. If you clip part of a character, (like a descender), the clipped character will have to be rebuilt. This takes time. Because of the difference between screen fonts and printer fonts, chances are that you will not be able to accurately clip the right characters unless you are running on the 128K ROMs or later and have fractional pixel widths turned on.

If you install a procedure to handle "Cancel Printing" requests and build in an option to "Pause" the printing process, beware of timeout problems. Communication between the Macintosh and the LaserWriter has to be maintained. If there is no communication for a period of time (up to two minutes), the printer will timeout and the print job will be terminated.

Lastly, if you encounter an error in the middle of a print loop, do not jump out; fall through and let the Print Manager terminate properly. For more information, see Handling Printing Errors. The most common error encountered is when there is no LaserWriter selected. It is a good idea to put up an alert asking the user to open the Chooser and select a printer when this error is encountered.