

#118: How to Check and Handle Printing Errors

See also: The Printing Manager

Written by: Ginger Jernigan

May 4, 1987

Updated:

March 1, 1988

This technical note describes how to check and properly handle errors that occur during printing with the high-level printing calls.

Most people are aware of the need for checking File Manager errors, Resource Manager errors, and the like, but sometimes Printing Manager errors get neglected; you should always check for error conditions while printing. This can be done by calling `PrError`. Errors returned by `PrError` will include any Printing Manager errors (and some AppleTalk and OS errors) that occur during printing.

The best place to start is with the code fragment on page 155 of *Inside Macintosh*, vol. II:

```
myPrPort := PrOpenDoc (prRecHdl, NIL, NIL); {open printing grafPort}
FOR pg := 1 TO myPgCount DO                {page loop: ALL pages of document}
  IF PrError = noErr THEN
    BEGIN
      PrOpenPage(myPrPort, NIL);             {start new page}
      IF PrError = noErr THEN
        MyDrawingProc (pg);                 {draw page with QuickDraw}
      PrClosePage (myPrPort);               {end current page}
    END;
  PrCloseDoc (myPrPort);
  IF prRecHdl^.prJob.bJDocLoop = bSpoolLoop AND PrError = noErr THEN
    BEGIN
      MySwapOutProc;                        {swap out code and data}
      PrPicFile (prRecHdl, NIL, NIL, NIL, myStRec); {print spooled document}
    END;
  IF PrError <> noErr THEN MyPrErrAlertProc; {report any errors}
```

Here are some error-handling guidelines:

- You should avoid calling `PrError` within your `PrIdle` procedure; errors that occur while it is executing are usually temporary and serve only as internal flags for communication within the printer driver—they are not intended for the application. If you absolutely must call `PrError` within your idle procedure, and an error occurs, never abort printing within the idle procedure itself. Wait until the last called printing procedure returns and then check to see if the error still remains. Attempting to abort printing within an idle procedure is a guarantee of certain death.

- If you detect that an error has occurred after the completion of a printing routine, just stop where you are, i. e. stop drawing. Proceed to the next print procedure to close any open calls you have made. For example, if you called `PrOpenDoc` and received an error, skip to the next `PrCloseDoc`. Or if you called `PrOpenPage` and got an error, skip to the next `PrClosePage` and `PrCloseDoc`. Remember that if some `PrOpen...` procedure has been called, then you must call the corresponding `PrClose...` procedure to ensure that printing closes properly and that all temporary memory allocations are released and returned to the heap.
- Do not raise any alerts or dialogs to report an error until the end of the print loop. At the end of the print loop, check for the error again; if there is no error assume that printing completed normally. If it's still there, you can raise an alert.

This is important for two reasons. First, if an alert is raised in the middle of the print loop, it can cause errors that will terminate an otherwise normal job. For example, if the printer is an AppleTalk printer, the connection can be terminated abnormally. While your alert is sitting there waiting for a response from the user, the driver is unable to respond to AppleTalk requests coming in from the printer. If the printer doesn't hear from the Macintosh within a short time period (30 seconds) then it will timeout, assuming that the Macintosh is no longer there. This results in the connection being broken prematurely causing another error that the application has to respond to.

The driver may also have already put up its own alert in response to the error. In this instance, the driver will post an error to let your application know that something went wrong and that it's time to abort printing. For example, when the driver detects that the version of Laser Prep that has been downloaded to the LaserWriter is different from the version that the user is trying to print with, the LaserWriter driver raises the appropriate alert telling the user that the printer was initialized with an incompatible version of the driver and gives the option of reinitializing. If the user chooses to cancel, the driver posts an error to let the application know that it needs to abort, but since the driver has already taken care of the error by putting up an alert, the error is reset to zero before the printing loop is complete. The application should check for the error again at the end of the printing loop and if it still indicates an error, it should raise an alert.