TECHNOTE 1097

Desktop Printing Revealed

By Dave Polaschek

devsupport@apple.com

CONTENTS

With the introduction of System 6 and MultiFinder in 1989, Apple introduced background printing. For the first time, a user could regain control of the Macintosh almost immediately after printing a document, while the printer driver processed the document in the background. Classic background printing has been far more robust than anyone may have imagined, and it's still the basis of much of today's printing technology, including desktop printing. Unfortunately for developers interested in supporting desktop printing, the interfaces to background printing have remained largely undocumented.

In 1994 the desktop printing architecture was introduced with the StyleWriter 1200 driver and LaserWriter 8.3. From its initial release to the present, desktop printing has existed as a collection of extensions and invisible applications. While this has not created a clear and simple picture for developers interested in supporting desktop printing, this technology shows every sign of thriving for a long time to come, so understanding its architecture is more vital than ever.

While the complete story is far from written, and things will certainly change again with the introduction of Rhapsody, this article will cover background printing for all pre-Rhapsody versions of the Mac OS. This includes the significant changes that will make Desktop Printing accessible to third party developers. If you've currently got a printer driver, this article describes everything you need to add support for desktop printing.

Classic Background Printing

The way early background printing worked was that printer drivers would save each page as a QuickDraw picture in the data fork of a temporary file (a spool file) in a special folder (the Spool Folder) within the System Folder. There was also information stored in the resource fork describing the page format, document name, and other job information, as well as offsets to the beginning of the data for each page. A special application (Backgrounder) launched by MultiFinder at startup time (in System 7, Backgrounder was incorporated into Finder) would see the document created by the printer driver and would launch PrintMonitor, which would run in the background, feeding the spool file to the printer driver.

PrintMonitor performs its magic by calling the printer driver in much the same way as an application would, except rather than your driver's 'PDEF' 0 resource getting called, its 'PDEF' 126 resource – which has the same format as the 'PDEF' 0 resource – is called. A special PrGeneral call is sent just after PrOpen has been called (before PrOpenDoc has been called), to provide the printer driver with the pointers to notification functions it will need to call. PrintMonitor then prints each page by replaying the stored page data to the driver. This method of background printing is still used by the Printing Manager today in two cases: when desktop printing isn't installed (or has been disabled) and when the Finder isn't running (usually because At Ease is running instead).

Getting to know classic background printing means you need to know the structure of its spool file. The resource fork of the spool file contains the following resources (note that all structures mentioned in this document have 68k alignment):

- 'PREC' 3 – the print record
- 'alis' -8192 – an alias to the driver that created the spool file
- 'ics#' 131 – the small icon to display for the spool file in the PrintMonitor window
- 'PREC' 124 – the printer name
- 'PREC' 126 – the job information

```
typedef {

        short   version;    // always 0

        short   flags;      // always 0

        short   numPages;   // total number of pages in the spool file

        short   numCopies;  // total number of copies for the spool file

        OSType  crtr;       // the creator type of the driver used to
                            // create the spool file

        Str31   appName;    // the application name used to print the
                            // spool file

} PREC126Record, **PREC126Handle;
```

- 'STR ' -8192 – the filename of the printer driver

The data fork of the spool file begins with a SpoolHeader structure, followed by the pages.

```
typedef struct {

          short   version;        // should always be 1

          long    fileLen;        // length of file including header

          long    fileFlags;      // should always be 0

          short   numPages;

          TPrint  printRecord;    // used only if PREC 3 can't be read

} SpoolHeader, *SpoolHeaderPtr, **SpoolHeaderHdle;

typedef struct {

          long    pictFlags;      // should always be 0

          Picture thePict;        // variable length

          long    pageOffset;     // offset to the beginning of this page's
                                  // PICT

} Page;
```

The spool file is created by the driver in the Spool Folder or PrintMonitor Documents folder within the System Folder (or in the current default desktop printer folder if Mac OS 8 desktop printing is active – you can find the correct folder with FindFolder and the kPrintMonitorDocsFolderType selector). The spool file has the name of the document being printed. If you're putting the file directly into the desktop printer folder, you'll need to append "(print)" to the filename (which means you may need to shorten the document name). Spool files that are being written have a type of '?job' and a creator of 'prmt'. Once the file is completely written, the driver changes the type to 'pjob'. When the version of Desktop Printing that supports third-party drivers is available, your driver also needs to send an Apple event to the Finder telling it that a new spool file was created (more on this below).

When PrintMonitor (or Desktop PrintMonitor) prints a job, it calls the driver's PrOpen routine and then calls PrGeneral with the structure shown below (see the DTPNotification.h header file for specifics). PrintMonitor then calls PrOpenDoc with a pIdleProc that the driver needs to call periodically. PrOpenPage and PrClosePage will get called for each page of the document, and the page will be printed to the driver.

```
// Function prototypes

typedef pascal void (*AsynchErrorNotificationProcPtr) (StringHandle string);

typedef pascal void (*EndNotificationProcPtr) ();

typedef pascal Boolean (*InForegroundProcPtr) ();

typedef pascal void (*StatusMessageProcPtr) (StringHandle string);


const short kPrintMonitorPrGeneral = -3;


typedef struct {

   short            iOpCode;          // kPrintMonitorPrGeneral
```

```
    short            iError;

    long             iReserved;        // 0 = PrintMonitor,
                                       // 1 = Desktop PrintMonitor

    THPrint          hPrint;

    short            noProcs;

    long             iReserved2;


// UPP to put up a notification

ErrorNotificationUPP  pASyncNotificationProc;


// UPP to take down the notification

EndNotificationUPP  pASyncUnnotifyProc;


// UPP to see if we are in the foreground

InForegroundUPP     pInForegroundProc;


// UPP to update the status message

// (available only with desktop printing that supports third parties)

StatusMessageUPP        pStatusMessageProc;

} TDesktopPrintingData;
```

When printing with background printing, both PrintMonitor and Desktop PrintMonitor will put a DialogPtr into the low-memory global ApplScratch just before calling PrOpenDoc. The driver should put status messages into the first item in that dialog using GetDialogItem and SetDialogItemText.

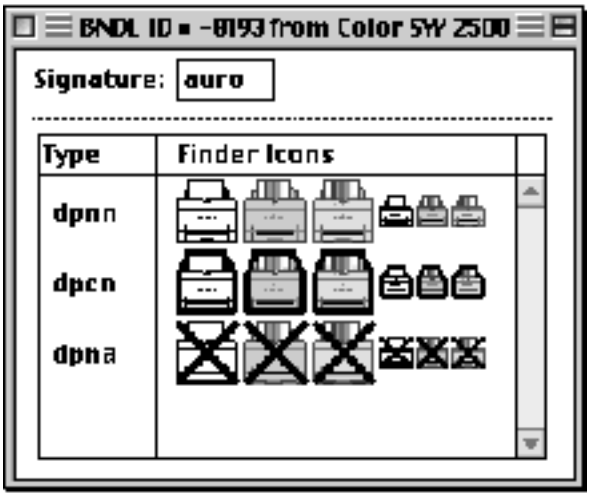
If the PrGeneral call says that printing is occurring from Desktop PrintMonitor (which is a faceless background application), no dialogs or alerts should be displayed. The one exception is that Desktop PrintMonitor patches StopAlert and ParamText. If you call ParamText and then StopAlert, the Finder will display an alert for you with the text you've set. However, the filter proc passed into StopAlert will not be called.

Desktop Printing Today

Desktop printing was introduced with the LaserWriter 8.3 and the StyleWriter 1200 printer drivers, and it uses much the same approach as classic background printing. Currently, desktop printing only supports Apple print drivers, but the additions described in this section are things you will need to incorporate into your drivers in order to work with desktop printing in the future.

The additional resources a driver needs to add to support desktop printing are the icons for the desktop printers. As a driver developer you'll need to supply a full 'BNDL' resource with your driver's creator. Beyond the types and icons you've probably already got in your driver (for the driver itself and any preferences file), you'll need to add icons for the types 'dpnn', 'dpcn', and 'dpna', which correspond to a "normal" desktop printer, a default desktop printer (with the heavy line around it), and an inactive (or unavailable) desktop printer. As an example, the bundle from the Color StyleWriter 2500 is shown below.

LaserWriter 8 has a slightly different scheme for setting the icons for desktop printers, in that it retrieves them from the PostScript printer description (PPD) file for a given printer.



The StyleWriter 2500 bundle and icons

When an Apple driver creates a spool file for desktop printing, it places the file in the same folder used by classic background printing. The Desktop Printing Extension will take care of moving the file to the desktop printer and beginning the process of actually printing the document. The resources defined immediately following are those added to the spool file when desktop printing is active:

- 'PINX' -8200 – the page index resource
- 'jobi' 1 – the print job information

```
typedef struct {

        short count;

        long pageoffset[1];  // The offset from the beginning of the file to
                             // the pageRecord (i.e., for the first page, it
                             // would be sizeof(SpoolHeader))

} PageIndex, *PageIndexPtr, **PageIndexHdl;

// Print priorities

#define kPrintJobUrgent     0x00000001

#define kPrintJobAtTime     0x00000002

#define kPrintJobNormal     0x00000003

#define kPrintJobHolding    0x00001003

typedef struct {

   short           firstPageToPrint;   // first page in the spool file to print

   short           priority;           // print priority
```

```
   short           numCopies;          // total number of copies
   short           numPages;           // total number of pages in the spool file
   unsigned long   timeToPrint;        // (when priority is kPrintJobAtTime)
   Str31           documentName;       // name of the document
   Str31           applicationName;    // the name of the application
   Str32           printerName;        // should match PREC 124
} PrintJobInfo, **PrintJobInfoHandle;
```

Another addition is a new way to change the default desktop printer. An application or driver can send an Apple event to the Finder as shown below. (Note that SendAEToFinder just sends the event to the Finder with an eventID of kFinderExtension.)

```
#define kDesktopPrinting   'dtpx'
#define kFinderExtension    'fext'
typedef struct {
      OSType    pfeCreator;
      OSType    extensionType;
      Str31     dtpName;
} SetDTPEvent;

OSErr SetDefaultDTP(StringPtr dtpName)
{
      OSErr         err=noErr;
      SetDTPEvent   myEvent;
      myEvent.pfeCreator = kDesktopPrinting;
      myEvent.extensionType = 'pfpr';
      pStrCpy(myEvent.dtpName, dtpName);
      err = SendAEToFinder((Ptr) &myEvent,sizeof(SetDTPEvent));
      return (err);
}
```

Desktop Printing Tomorrow

With the introduction of Mac OS 8, desktop printing will no longer be a separate extension. It will be integrated into the Finder and therefore available in most cases. A user can still disable desktop printing by disabling the Desktop PrintMonitor and Desktop Printer Spooler in Extensions Manager, though. In a future

system software release (post MacOS 8.0), changes will be made to make it possible for third parties to integrate support. These changes are also being made to the previous versions of desktop printing so third parties (that's you!) can use desktop printing on systems which have not been upgraded to Mac OS 8.

Desktop printing installs a Gestalt selector to tell you if third-party drivers can be supported. If desktop printing is available, but this selector does not report that third-party drivers are supported by desktop printing, you will need to use the methods described in the "Classic Background Printing" section above.

```
#define kGestaltPFEFeatures  'dtpf'

#define kThirdPartySupport   0x00000004

Boolean ThirdPartyDriverSupported(void)

{

    long  response;

    Boolean result = false;

    OSErr err = Gestalt(kGestaltPFEFeatures,&response);

    if (err == noErr)

    result = !!(response & kThirdPartySupport);

    return result;
}
```

When non-Apple drivers are supported by desktop printing, your driver needs to write your spool files directly to the desktop printer folder with the type of '?job'. When you are done spooling, you need to change the file's type to 'pjob'. The driver can determine the current default desktop printer folder, and many other things, by calling the Gestalt routine with the desktop printing extension selector. The selector is 'dtpx', and the information is returned to you in a handle. When you're done with the handle, do not call DisposeHandle on theDTPList and the GestaltPFEInfoHdle.

```
typedef struct {

        short   vRefNum;        // vRefNum of the desktop printer folder

        long    dirID;          // Directory ID of the desktop printer

        Str31   dtpName;        // Name of the desktop printer folder

        OSType  driverType;     // Driver's creator

        Boolean isDefault;      // Is this the default desktop printer?

        Str32   printerName;    // Network name of the printer(only for
                                // LaserWriter 8.4 desktop printers)

        Str32   zoneName;       // Zone of the printer (only for LaserWriter
                                // 8.4 desktop printers)
} DTPInfo, *DTPInfoPtr;

typedef struct
{
    long    structversion;      // Version of this structure
```

```
    short   numDTPs;            // Number of desktop printers in the list

    Handle  theDTPList;

} GestaltPFEInfo, **GestaltPFEInfoHdle;
```

When you're done writing the spool file, you need to send a Core Apple event to the Finder with the following direct object key:

```
typedef struct {

        OSType    pfeCreator;    // Set this to 'dtpx'

        OSType    pfeEventType;  // Set this to 'pfsc'

        FSSpec    dtpSpec;       // The file spec of the desktop printer where
                                 // the new spool file was added

} SyncDTPEventData;
```

There are also added calls for this version of desktop printing. Specifically, there are three new PrGeneral selectors that a driver needs to support: kIsSamePrinterInfo, kGetPrinterInfo, and kSetDefaultPrinterInfo. These selectors enable the desktop printing extension to decide which of the driver's desktop printers is the current default printer, to determine whether it needs to create a new desktop printer, and to inform the driver that a desktop printer has been selected as the default. The structures you'll need to use to support these selectors are shown below.

```
// DTP printer types

enum { kSerial, kAppleTalk, kTCPIP, kUnknown };

enum { kPrinterPort, kModemPort };

enum {

   kGetPrinterInfo = 23,

   kIsSamePrinterInfo = 24,

   kSetDefaultPrinterInfo = 25

};


typedef struct {

     short  port;               // kPrinterPort or kModemPort

} SerialPrinterInfo;

typedef struct {

     Str32   nbpName;

     Str32   nbpZone;

     Str32   nbpType;
```

```
} AppleTalkPrinterInfo;

typedef struct {

     Str255  TCPIPAddress;

} TCPIPPrinterInfo;

typedef struct {

     Str31   dtpDefaultName;    // Default name to be used for the desktop
                                // printer. The desktop printing extension
                                // will add a suffix if there's a conflict
                                // with other desktop printers.

     short   printerType;


     // Info specific to each class of printers

     union {

        SerialPrinterInfo       serialInfo;

        AppleTalkPrinterInfo    appleTalkInfo;

        TCPIPPrinterInfo        TCPIPInfo;

     } u;

// Optional driver-specific information can be appended here.

} DTPPrinterInfo, **DTPPrinterInfoHandle;


typedef struct {

   short            iOpCode;

   short            iError;

   long             iCommand;

   DTPPrinterInfoHandle printerInfo;

} TPrinterInfoPrGeneralData;
```

When your driver is selected in the Chooser, desktop printing will call your driver via a series of PrGeneral calls. First, you'll get called with the kIsSamePrinterInfo selector for each of the desktop printers created by your driver in order to determine which is the currently selected printer. Your driver responds by filling in the iError field of the TPrinterInfoPrGeneralData record. If the printer that your driver thinks is current matches the information passed in with the kIsSamePrinterInfo selector, the driver responds by setting the iError field to noErr. If it's not a match, the driver sets the iError field to -1.

If the printer selected in the Chooser is not among the desktop printers owned by your driver, desktop printing will create a new desktop printer and call PrGeneral with the kGetPrinterInfo selector to get the information for the selected printer. At this point, your driver should resize the printerInfo handle and fill in the printer information. You need to fill in the dtpDefaultName field with the name you'd like to see a desktop printer created with. Also note that your driver can append as much (or as little) extra printer information as you'd

like. Once you've returned the information, the desktop printing extension will save this information into the desktop printer.

If a user selects one of your desktop printers via the Set Default Printer menu item in the Printing menu (which appears in the Finder when you've clicked on a desktop printer), your driver will get a PrGeneral call with the kSetDefaultPrinterInfo selector. When you receive this call, you should change any internal settings your driver maintains so that the printer pointed to by the DTPPrinterInfo you received is now the currently selected printer for your driver.

The Future Is Now

If you've currently got a driver that supports classic background printing, your best bet for future compatibility is to add support for desktop printing. However, if you're just starting to tackle background printing now, while you could implement support for Mac OS 8 desktop printing only, we strongly recommend that you support the current desktop printing architecture and classic background printing, as well. That way your users will have a more consistent experience when printing, regardless of which version of the Mac OS they're running.

Acknowledgments

THANKS to reviewers Alan Beck, Rich Blanchard, Paul Danbold, Hueii Huang, Ingrid Kelly, and Donna Lee. And thanks to Jimmy Buffett for the soundtrack.

Send feedback to devsupport@apple.com
Updated: 9-June-97