Technotes

Download  Acrobat file (K)	Download  AppleWorks file (45K)

Querying PostScript Printers at dtp Creation Time the QuickDraw GX Way

---

Technote 1057 **JULY 1996**

For some time now, Apple developers have asked how they can query their PostScript printers at dtp creation time using the same methods as the LaserWriter GX printer driver. This Technote discusses how the LaserWriter GX driver does it at query time and provides you with the printing messages you need to override in order to achieve the same functionality.

You may need to change or add to the code in this Note. For example, if you want to query for the number of paper trays or query for an envelope tray, you will need to modify this code, as explained later in this Note.

This Note is intended for QuickDraw GX printer driver developers who wish to query their PostScript printer at dtp creation time.

CONTENTS

Important for all Apple Printing and Graphics Developers:

The information in this Technote is still relevant up to and including Mac OS 7.6 with QuickDraw GX 1.1.5. Beginning with the release of Mac OS 8.0, however, Apple plans to deliver a system which incorporates QuickDraw GX graphics and typography **only**. QuickDraw GX printer drivers and GX printing extensions will **not** be supported in Mac OS 8.0 or in future Mac OS releases. Apple's goal is to simplify the user experience of printing by unifying the Macintosh graphic and printing architectures and standardizing on the classic Printing Manager.

For details on Apple's official announcement, refer to </technotes/gxchange.html>

Necessary Requirements for Querying Your Printer

In order to query your printer at dtp creation time, you must:

- adhere to PostScript calling conventions, as documented in Adobe's PostScript Language Reference Manual
- send valid PostScript query code -- again, as specified in Adobe's PostScript Language Reference Manual
- use two-way synchronous device communications with your printer. Currently, GX does not have messages that support three-way communications

Note:
The method presented here is only good for two-way synchronous device communications. At some point in the future, GX may add new printing messages to support three-way communications via OpenTransport. The third channel in this configuration is the `status' channel.

Implementing Message Overrides

In order to query your printer at dtp creation time, you need to override several QuickDraw GX messages. The LaserWriter GX driver overrides gxInitialize, gxShutdown, gxDefaultDesktopPrinter, gxJobStatus, gxJobIdle, and gxPostScriptQueryPrinter. If you want to duplicate the LaserWriter GX functionality, you need to override these same messages in your GX driver.

gxInitialize

QuickDraw GX sends the gxInitialize message at the start of a new printing job. You should override gxInitialize so that your driver may use global data.

Within this override, you need to set up an A5 world for your globals (GX 1.1.3 and earlier). The LaserWriter GX driver does this by means of a call to NewMessageGlobals. This function is documented in *Inside Macintosh: QuickDraw GX Environment and Utilities* , page 6-17.

gxShutdown

QuickDraw GX sends the gxShutdown message at the end of a print job. You should override gxShutdown to clean up your driver's global data.

Within this override, you need to dispose of your globals and deallocate the memory used by your driver for globals. The LaserWriter GX driver does this by means of a call to DisposeMessageGlobals. This function is documented in *Inside Macintosh: QuickDraw GX Environment and Utilities* , page 6-18.

gxDefaultDesktopPrinter

QuickDraw GX sends the gxDefaultDesktopPrinter message when the user creates a new desktop printer with the Chooser. You should override gxDefaultDesktopPrinter to open the connection to the currently selected printer during dtp creation time. The LaserWriter GX override of this message opens a connection to the printer by means of the GXOpenConnection message and then queries the printer to find out what type of printer is currently selected.

The LaserWriter GX override of gxDefaultDesktopPrinter also puts up a dialog box, providing the user with one of two options: either use the automatic setup for the printer (i.e., gets the paper source and imaging options), or cancel out. You can expand this dialog and the printer query, for instance, to get the number of paper trays from your printer. The various query comments that you can check for are documented on pages 690-696 of *PostScript Language Reference Manual* , Second Edition.

gxJobStatus

QuickDraw GX sends the gxJobStatus message to display the current status of a print job during spooling and despooling. You should override gxJobStatus to show status in your dtp window. The LaserWriter GX driver does this by means of the GXWriteStatusToDTPWindow message.

Note:
When support for three-way communications is added to QuickDraw GX, more than likely you won't have to override this message.

gxJobIdle

QuickDraw GX sends the gxJobIdle message when a printer driver, a printing extension or an application calls the GXJobIdle function to release idle time. You should override gxJobIdle to idle the printer. The LaserWriter GX driver overrides this message to show dialog actions during the dtp creation query and to give idle time to the automatic setup dialog.

gxPostScriptQueryPrinter

QuickDraw GX sends the gxPostScriptQueryPrinter message to gather configuration information before imaging begins. You should override gxPostScriptQueryPrinter to perform the actual query of the printer. In this message override, you need to add your PostScript queries and then save the results to the dtp by means of GXWriteDTPData.

Additional Dialog Handling Functions

This section introduces you to two optional dialog handling functions.

DisposeQueryDialog

This function disposes of the automatic setup dialog used for the queries in the message override of gxDefaultDesktopPrinter. The LaserWriter GX driver disposes of the dialog by means of DisposDialog. DisposDialog is documented in *Inside Macintosh: Macintosh Toolbox Essentials* , page 6-120.

LoadQueryDialog

This function loads the automatic setup dialog used for the queries in the message override of gxDefaultDesktopPrinter. The LaserWriter GX driver loads the dialog by means of GetNewDialog. GetNewDialog is documented in *Inside Macintosh: Macintosh Toolbox Essentials* , page 6-113.

The Query Code

The following query code shows the message overrides for Apple's specific line of PostScript printers. You may have to make changes to this code in order to adapt it to your driver's needs.

Globals

```
// ------------------------------------------------------
// GLOBALS
// ------------------------------------------------------

// variables for performing our query

enum
        {
        kMaxQueryTimeBeforeStatus= 60   // ticks before we put up
                                        // status
        };

Boolean         gNormalStatus= true;    // should the status query go
                                        // through normally?
long                    gQueryIdle= 0;  // time before presenting a
                                        // dialog during query
DialogPtr               gQueryDialog= nil;      // dialog used for status
                                                // during query
```

```
short           gQueryItemHit= 0;       // item that was hit in the
                                        // querying dialog.
```

Message Overrides

```
// ------------------------------------------------------
// MESSAGE OVERRIDES
// ------------------------------------------------------

// DriverInitialize: This override lets us use global data.

OSErr           DriverInitialize(void)
{
                OSErr           anErr;

                anErr = NewMessageGlobals(A5Size(), A5Init);
                return(anErr);


}// DriverInitialize

// ------------------------------------------------------

// DriverShutdown: This override cleans up our global data.

OSErr           DriverShutdown(void)
{
        DisposeMessageGlobals();
        return(noErr);

}// DriverShutdown

// ------------------------------------------------------

// DriverDefaultDesktopPrinter: This override is invoked when a
// new desktop printer is created.

OSErr           DriverDefaultDesktopPrinter(Str31 dtpName)
/*
        This call is made to setup a newly made desktop printer.

        We override this message to perform a query during creation time
        to determine the type of the printer.
*/
{
        OSErr           anErr;
        Handle          configHandle;
        ResType         commType;

        gQueryDialog = nil;

        // make the printer first
        anErr = Forward_GXDefaultDesktopPrinter(dtpName);
        nrequire(anErr, DefaultDesktopPrinter);

        // initialize the status informational state
        gQueryIdle = TickCount() + kMaxQueryTimeBeforeStatus;

        // determine communications type
        anErr = GXFetchDTPData(dtpName, gxDeviceCommunicationsType,
                        gxDeviceCommunicationsID, (Handle*)&configHandle);
        nrequire(anErr, FetchCommType);
        commType = **(ResType**)configHandle;
        DisposeHandle((Handle) configHandle);
```

```
// query PAP devices for configuration information
if (commType == 'PPTL')
        {
        short           curResFile = CurResFile();
        short           hit;

        // we don't yet have a status dialog, get one
        LoadQueryDialog();

        if (gQueryDialog)
                {
                Str32           printerDeviceName;
                Handle          theHandle;
                short           theKind;
                Rect            theRect;

                // set up keyboard for cancel button and ok button
                SetDialogCancelItem(gQueryDialog, cancel);
                SetDialogDefaultItem(gQueryDialog, ok);

                // fill in the name of the printer we are querying
                GXGetPrinterName(GXGetJobOutputPrinter(GXGetJob()),
                                printerDeviceName);
                ParamText(printerDeviceName, "\p", "\p", "\p");

                // setup our user items
                SetUserItem(gQueryDialog, iQueryIcon, (ProcPtr)
                                DrawLaserWriterIcon);
                SetUserItem(gQueryDialog, iTitleUnderline, (ProcPtr)
                                DrawRectangle);
                SetUserItem(gQueryDialog, iStatusOutline, (ProcPtr)
                                DrawRectangle);

                // and reveal all to the user
                ShowWindow(gQueryDialog);
                // We can take status text now
                gNormalStatus = false;

                // make the cursor an arrow

                InitCursor();


                // if the user wants to, do a query
                ModalDialog(nil, &hit);

                if (hit == cancel)
                {
                        anErr = gxPrUserAbortErr;
                        DisposeQueryDialog();
                }
                else
                {
                        // disable the setup button -- because we
                        // have started the query
                        GetDItem(gQueryDialog, ok, &theKind,
                                &theHandle, &theRect);
                        HiliteControl((ControlHandle) theHandle,
                                255);
                }
        }

        if (anErr == noErr)
                {
```

```
                // open the connection in order to query the printer
                anErr = Send_GXOpenConnection();

                if (anErr == noErr)
                        {
                        // here we do the actual query -- if we get
                        // an error we should allow the
                        // user to select the printer type directly
                        Handle          imageDataHdl;

                        imageDataHdl = NewHandle(0);
                        anErr = MemError();
                        if (anErr == noErr)
                        {
                                anErr = send_GXSetupImageData(imageDataHdl);
                                DisposeHandle(imageDataHdl);
                        }

                        // all done with the printer connection now
                        (void) Send_GXCloseConnection();
                }
        }

        else
                anErr = noErr;
        }

        // No more status text, thanks.
        gNormalStatus = true;



// if during the query we put up a dialog, get rid of it
if (gQueryDialog)
        {
        Handle                  theHandle;
        short                   theKind;
        Rect                    theRect;
        long                    ignore;

        // Get the OK or cancel button--whichever was selected
        if (anErr == noErr)
                GetDItem(gQueryDialog, ok, &theKind, &theHandle,
                        &theRect);
        else
                GetDItem(gQueryDialog, cancel, &theKind, &theHandle,
                        &theRect);

        // Our jobIdle rtn stores the item hit (if any) in
        // gQueryItemHit. We do it this way so that we don't
        // invert the ok or cancel buttons if the user already did
        // that.  Instead, we only auto-invert if we auto-
        // dismissed the dialog.  If gQueryItemHit is 0, the user
        // didn't dismiss the dialog, we did.

        if (gQueryItemHit == 0)
        {
                HiliteControl((ControlHandle) theHandle, inButton);
                Delay(10, &ignore);
                HiliteControl((ControlHandle) theHandle, 0);
        }

        DisposeQueryDialog();
        // if we have a query error, we can just ignore it.  This
        // is not fatal to the DTP creation process
```

```
                if (anErr != noErr)
                {
                        short           curResFile = CurResFile();

                        UseResFile(GXGetMessageHandlerResFile());
                        StopAlert(kQueryFailedAlert, nil);
                        UseResFile(curResFile);

                        // we could allow the user to select the printer
                        // type themselves but there are UI issues here.
                        anErr = noErr;
                }
        }
FetchCommType:
DefaultDesktopPrinter:
        return(anErr);

}// DriverDefaultDesktopPrinter


// ------------------------------------------------------
// DriverJobStatus: This call is made to status a printer.

OSErr           DriverJobStatus(gxStatusRecord *pStatus)
/*
        This call is made to status a printer.

        We override this to perform nothing (ie, no status dialog) during
        our create printer query.
*/
{
        OSErr           anErr = noErr;

        if (gNormalStatus || !gQueryDialog)
                {
                anErr = Forward_GXJobStatus(pStatus);
                nrequire(anErr, JobStatus);
                }
        else
                {
                gxDisplayRecord theDisplay;

                theDisplay.hPicture = nil;
                anErr = Send_GXWriteStatusToDTPWindow (pStatus,
                        &theDisplay);
                nrequire(anErr, GetStatusString);

                if ((theDisplay.useText) && (gQueryDialog))
                        {
                                Handle                  theHandle;
                                short                   theKind;
                                Rect                    theRect;
                                GrafPtr                 oldPort;

                                // Now, draw the updated status message in
                                // the dialog.  We don't use SetItemText
                                // here, because it uses more memory than
                                // TextBox.

                                GetPort(&oldPort);
                                SetPort(gQueryDialog);
                                GetDItem(gQueryDialog, iQueryStatusField,
                                        &theKind, &theHandle, &theRect);
                                TextBox(&theDisplay.theText[1],
                                        theDisplay.theText[0], &theRect,
```

```
                                        teFlushDefault);
                                SetPort(oldPort);
                        }
        if (theDisplay.hPicture)
                DisposeHandle(theDisplay.hPicture);

                }

// FALL THROUGH EXCEPTION HANDLING
GetStatusString:
JobStatus:

        return(anErr);

}// DriverJobStatus

// ------------------------------------------------------
// DriverJobIdle: This call is made to idle a printer connection.

OSErr           DriverJobIdle(void)
/*
        This call is made to idle a printer connection.

        We override this to do dialog actions during our create printer query.
*/
{
        OSErr                                   anErr = noErr;

        anErr = Forward_GXJobIdle();
        nrequire(anErr, JobIdle);

        if ((gQueryDialog != nil) && (gNormalStatus == false) &&
                (TickCount() > gQueryIdle))
        {
                // conduct the dialog if it's open
                EventRecord                     theEvent;
                DialogPtr                               dPtr;
                short                                   hit;

                // setup our user items (the code may move each time)
                SetUserItem(gQueryDialog, iQueryIcon, (ProcPtr)
                        DrawLaserWriterIcon);
                SetUserItem(gQueryDialog, iTitleUnderline, (ProcPtr)
                        DrawRectangle);
                SetUserItem(gQueryDialog, iStatusOutline, (ProcPtr)
                        DrawRectangle);

                WaitNextEvent(everyEvent, &theEvent, 0, nil);

                if (IsDialogEvent(&theEvent))
                {
                        Ptr     oldD3;

                        DialogSelect(&theEvent, &dPtr, &hit);

                        // The StdFilterProc has a bug that trashes D3
                        oldD3 = GetD3();
                        StdFilterProc(dPtr, &theEvent, &hit);
                        StuffD3(oldD3);

                        // This is so that we don't double-flash buttons
                        // when they're pressed.  See
                        // DriverDefaultDesktopPrinter.

                        if ((hit == cancel) || (hit == ok))
```

```
                        gQueryItemHit = hit;
                    if (hit == cancel)
                        anErr = gxPrUserAbortErr;
                }
        }

// FALL THROUGH EXCEPTION HANDLING
JobIdle:

        return(anErr);

}// DriverJobIdle

// ------------------------------------------------------
// DriverQueryPrinter: This call is made to query the printer.

OSErr           DriverQueryPrinter(long * queryResult)
{
        OSErr           anErr;

        anErr = Forward_GXPostScriptQueryPrinter(queryResult);
        nrequire(anErr, ForwardFailed);

        // Some printers (e.g. LW 16/600) close the connection if the
        // Chooser querying code initializes the printer and executes an
        // exitserver. Since the exitserver is unnecessary for the
        // Chooser query, when we do the Chooser queries, we clear the
        // query result if it was gxIntializePrinter or gxResetPrinter.
        // (For reset we also wait until the printer has reset itself.)
        // Since we never return gxResetPrinter or gxIntializePrinter
        // from the gxPostScriptQuery message (for the Chooser), the PS
        // driver will never send gxPostScriptInitialize. Therefore, an
        // exitserver will never happen during the Chooser queries.

        if (gQueryDialog && ((*queryResult == gxResetPrinter) ||
                (*queryResult == gxIntializePrinter)))
        {
                if (*queryResult == gxResetPrinter)
                // Wait until the printer is ready again
                        nrequire(anErr = Send_GXPostScriptResetPrinter(),
                                ResetPrinterFailed);

                *queryResult = gxPrinterOK;
        }

        if ( (anErr == noErr) && (*queryResult != gxFilePrinting) )
        {
                                // Do whatever special querying you need to
                                // do, and then save the results to the
                                // desktop printer using GXWriteDTPData.
                                // You can get the data from the desktop
                                // printer by means of GXFetchDTPData
        }

ResetPrinterFailed:
ForwardFailed:
        return(anErr);

}// DriverQueryPrinter
```

Dialog Functions

```
// ------------------------------------------------------
/*
                DisposeQueryDialog: This call disposes of the dialog that
```

```
                we use for the DefaultDesktopPrinter queries.
*/

void DisposeQueryDialog(void)
{
        if (gQueryDialog)
        {
                // Dispose of the query dialog.

                DisposeDialog(gQueryDialog);
                gQueryDialog = nil;
        }
}


// ----------------------------------------------------
/*
                LoadQueryDialog: This call loads the dialog that we
                use for the DefaultDesktopPrinter queries.
*/

void LoadQueryDialog()
{
        short           curResFile = CurResFile();
        // Load the query dialog.

        UseResFile(GXGetMessageHandlerResFile());
        gQueryDialog = GetNewDialog(kQueryStatusDialog, nil,
                                    (WindowPtr)-1);
        UseResFile(curResFile);
}
```

Summary

The code you need to query your printer for information at dtp creation time is concise and straightforward. The most complicated part of implementing this code for your driver is deciding what PostScript query comments you want to use and figuring out what print messages you want to override.

Further References

- *PostScript Language Reference Manual, Second Edition* , Addison-Wesley.
- *Inside Macintosh: QuickDraw GX Printing Extensions and Drivers: Chapter 4*
- *Inside Macintosh: QuickDraw GX Environment and Utilities.*
- *Inside Macintosh: Macintosh Toolbox Essentials.*
