Technical Note PR14
Dictionary Downloading

CONTENTS

This technical note discusses a method for downloading PostScript dictionaries automatically using the LaserWriter driver. It will also provide the format and use of the PREC(103) resource. It will also describe some problems with the now obsolete PREC(201) resource. If you are using PostScript dictionaries, or either of these resources, you should definitely read this note.

[Apr 01 1991]

Introduction

Although many picture comments have been added to support the features of PostScript that are missing from Quickdraw, many developers have still taken to sending PostScript directly from their applications. As the use and complexity of this PostScript code increases, more and more developers are finding it necessary to define and use their own PostScript dictionaries. PostScript dictionaries are basically collections of variables and procedures that can be predefined, and accessed later. They are used to prevent conflicts between the symbols used by applications and those used by system software (such as the LaserWriter driver's LaserPrep dictionary). Unfortunately, because of the LaserWriter drivers habit of using the PostScript "save" and "restore" operators, there are problems with keeping a PostScript dictionary defined. PostScript definitions made by code sent with the print job (i.e., sent between the calls to `PrOpenPage/PrClosePage`) will be lost the first time the LaserWriter driver calls "restore." There are a couple of solutions to this problem, but one that hasn't been documented before involves the use of the PREC(103) resource. If the LaserWriter driver finds a resource of type PREC with an ID of 103, it will download the PostScript code to the LaserWriter **before** performing the initial "save" operation. This means that any definitions made by the PostScript code stored in the PREC(103) resource will remain defined for the duration of the print job, independent of the LaserWriter driver's calls to save and restore.

Back to top

Caveats

The PREC(103) method is a great way to get a dictionary downloaded at print time. Unfortunately, this does not solve the problem for using dictionaries in export files like PICT. If you insert PostScript code into Quickdraw pictures, the system is not smart enough to record the PREC(103) code into the picture. Instead, you must record the dictionary using the standard PostScript picture comments (defined in Technical Note #91, Optimizing for the LaserWriter-PicComments). You should also use the appropriate PostScript structuring comments as defined by the *Adobe Document Structuring Conventions* . If you use the Adobe comments correctly, an application that is importing your picture will have the option of parsing for the procedure set comments, and extracting the dictionary definition to be placed in a PREC(103) resource at print time.

The next caveat concerns the use of multiple PREC(103) resources. At `PrOpenDoc` time, the LaserWriter driver makes one `GetResource` call to load the resource of type PREC with an ID of 103. Because the call is a `GetResource` call (instead of `Get1Resource`), the PREC can be stored in any open resource file. To avoid any conflicts, the resource should be stored in the resource fork of your application, or in the document file that is referencing the PostScript dictionary. Because the `GetResource` call is only made once, only the first PREC resource found by `GetResource` will be used. Any other PREC(103) resources will be ignored. As long is this resource is only used by applications, there is no problem since there can only be one application active at any particular time. If the resource is used by other elements of the system (i.e., desk accessories, drivers, INITs), you can easily run into the problem of your PREC resource being ignored. The best solution to this problem is to only use the resource from within an application.

Since the PREC(103) resource is considered part of the print job, the definitions it makes are lost when the job ends (i.e., when the LaserWriter receives `EndOfFile` from the Macintosh). Because of this, the code you place in the PREC(103) resource should not attempt changing any persistent parameters in the printer. The means avoiding the PostScript "exitserver" operator. You should also avoid calling other routines that reset the current state of the printer (i.e., initclip, initgraphics, etc.). Use of these operators will have a serious effect on Quickdraw operations that may be present in the print job.

When the PREC(103) resource was originally introduced, it had a cousin called PREC(201). PREC(201) was similar to the PREC(103) resource in that it allowed PostScript to be downloaded to the printer before the actual print job. The difference between the two resources was that the PREC(201) resource downloaded the PostScript code at the same time that it downloaded the LaserPrep dictionary, outside of the PostScript "server loop." Because of this, any definitions made by the code in the PREC(201) resource would remain after the current job. Like the LaserPrep dictionary, the dictionary downloaded in PREC(201) would remain until the LaserWriter was rebooted (i.e., powered off then on again). Although this feature was useful in some situations, it did have its problems. Not the least of which was the valuable printer memory consumed by the dictionary that was downloaded. Since the dictionary remained after the job that required it, subsequent jobs had less memory available to them. The only way to reclaim the memory was to reboot the printer, and this was not obvious to naive users. The other problem was introduced when background printing became available. With background printing enabled, the LaserWriter driver could no longer count on the PREC(201) resource always being available. Since you could no longer store the resource in the LaserWriter driver (because of the LaserWriter driver being MultiFinder compatible - see *Learning To Drive* for more information), it has to be stored in a separate resource file. This made it virtually impossible for the LaserWriter driver to find the resource when it was required. For this reason, the PREC(201) resource is only downloaded when background printing is turned off.

Needless to say, we don't recommend the use of features that only work in certain situations, so the PREC(201) resource is now considered unsupported and obsolete. If you are using the PREC(201) resource, you should be able to revise your application to use the PREC(103) resource instead, with only a small performance penalty. On the bright side, the PREC(201) resource will continue to be supported in the foreground through the 7.0 version of the LaserWriter driver, and most likely, until the new printing architecture becomes available, giving you plenty of time to revise....

Back to top

Implementation

The PREC(103) resource can be implemented by simply creating the resource with ResEdit or Rez, and then storing it in an open resource file at print time. In the case of ResEdit, you should create a new resource of type PREC with an ID of 103. You should then open the new resource using the resource template for string (`'STR '`) resources. You can then type your PostScript code directly into the resource.

If you would rather keep your PREC definition as a Rez source file with the rest of your project, you can do this by simply defining the PREC resource type at the top of the file, followed by an instance of the PREC resource. Consider the following Rez source code:

```
    /* First the resource type definition: */
    type 'PREC' {
        string;
    };

    /* Now the real resource definition: */
    resource 'PREC' (103) {
        "userdict /mydict 50 dict def";
```

We begin by defining the resource type as being a string. We then define an instance of the resource with an ID of 103. Finally, we define the contents of the resource. The PostScript code above basically defines a dictionary named mydict within the userdict dictionary. The mydict dictionary is defined as having a maximum of 50 elements. Consult the PostScript Language Reference Manual for more information concerning legal operations on dictionaries.

Back to top

Conclusion

The PREC(103) is a simple, efficient way to download a PostScript dictionary at print time. It does not solve the problem of exporting PostScript that references a dictionary into file formats such as PICT, but it can help. Applications can be revised to extract PostScript dictionary definitions from files such as PICT and download them at print time using the PREC(103). It should be noted however that this is not automatic, the application must parse the picture to get this functionality. Finally, the PREC(201) resource can only be supported when background printing is disabled, so it is now considered obsolete, and use of it is unsupported.

Back to top

References

PostScript Language Reference Manual, Adobe Systems Inc.

Adobe Document Structuring Conventions, Adobe Systems Inc.

LaserWriter Reference Manual, Addison-Wesley

Back to top

Downloadables

	Acrobat version of this Note (K).	Download