TUG-GHOSTSCRIPT

RICHARD KOCH

1. Ghostscript

This is the portion of the TUG documentation which deals with making the Ghostscript package. A clean package file will contain

- "About" folder with public documentation explaining how the package was constructed
- "Binaries" folder with empty subfolders "10.5", "10.6" and the file "liposcript"
- Files "buildPackage.sh" and "buildTeXfonts.sh"
- Files "background.jpg", "License.rtf", "ReadMe.rtf", and "Welcome.rtf"
- "scripts" folder containing files "postinstall" and "setloginpath"
- "source" folder containing "ghostscript-fonts-std-8.11.tar" and a folder of fonts named "fonts". The tar folder contains a packed version of these fonts. (This folder is not needed in the current package because Ghostscript now contains the 35 standard fonts. It will eventually be removed.) The "source" folder also contains a folder named "TeXfonts" which in turn contains files named "Fontmap" and "Fontmap.cmr" and a folder named "Bruno-Arnold". This "BA" folder contains correspondence about a problem in earlier distributions of the Ghostscript package which is fixed in this version. For details, see later sections of this document.
- "TUG" folder with this document

To construct the package, download the latest Ghostscript tar file from `http://www.ghostscript.com/` and put the file in the "source" folder. Then unzip this file; the resulting folder is the source code for Ghostscript.

2. OverView

This folder contains an "About" folder giving an overview of the construction of the Ghostscript package. You may want to read that document, because otherwise the compilation steps below will seem unnecessarily baroque.

3. Leopard binaries

Build Ghostscript on Leopard Intel and Leopard PPC. On each of these operating systems, built a Ghostscript binary with X11 support, and a binary without X11 support. Before building, make sure that Apple's X11 package is installed on both systems.

We'll use Intel as an example. Move a copy of the Ghostscript source code to the Intel Leopard system. Using Terminal, switch directories to /usr and rename local to local-temp. Then in the Ghostscript source code type

```
make clean
LDFLAGS="-L/usr/lib" ./configure
make
sudo make install
```

Use the Finder's "Go" menu to go to /usr/local/bin and copy gs to the desktop. Rename it gs-X11-Intel. Next go to /usr and

```
sudo rm -R local
sudo mv X11R6 X11R6-temp
```

If there is a folder named X11, rename it as well. Then rebuild Ghostscript as above, move gs in /usr/local/bin to the desktop, and name it gs-noX11-Intel.

Finally, clean up /usr by moving /usr/local-temp back to /usr/local, moving X11R6-temp to X11R6, and similarly for X11 if it existed.

Repeat this entire process on PowerPC Leopard, calling the resulting binaries gs-X11-PPC and gs-noX11-PPC.

Finally, move the four resulting binaries to the Binaries/10.5 directory of this MacTeX Ghostscript build tree.

4. Snow Leopard binary

Build two binaries on Snow Leopard (Intel only, of course), one with X11 support and one without. Note that X11 is installed by default on Snow Leopard and Lion, but not on Mountain Lion, so the 64 bit binary without X11 support is needed.

Repeat the steps in the Leopard build described earlier. In the actual compile step on Snow Leopard, the LDFLAGS flag is not needed, so the compile instructions become

```
make clean
./configure
make
sudo make install
```

Call the resulting binaries gs-X11-64 and gs-noX11-64. Move the resulting binaries to the Binaries/10.6 directory of this MacTeX Ghostscript build tree.

5. Putting It All Together

On whatever platform you use to build install packages, switch to the MacTeX build tree's Ghostscript folder; change to the Binaries directory, and type

```
sh liposcript
```

This will lipo the various binaries together. The script will automatically throw away old copies and then create new copies of gs-noX11, gs-X11, gs-X11-64Bit and gs-noX11-64Bit. Leave them in the directories where the liposcript put them. The first two binaries are universal with 32 bit code for PPC and Intel. The last two binaries are universal with Intel 32 bit and 64 bit code. The 32 bit code is required because Snow Leopard runs on early Intel machines without 64 bit processors.

Next edit the file "buildPackage.sh" to reflect the current version of Ghostscript. Currently this file contains "9.05", referring to folders installed by Ghostscript. Change this number appropriately.

Next make sure the latest TeX Live is installed on this machine and type

```
sh buildTeXfonts.sh
```

This step will create a folder named "fonts" in "source/TeXfonts/ containing symbolic links to the pfb font files in this latest TeX Live distribution.

Next change directory to /usr and rename /usr/local to /usr/local-temp. Copy the Ghostscript source folder to your machine and install Ghostscript using

```
make clean
./configure
make
sudo make install
```

In this step we are not interested in the binaries, but instead in the support files for Ghostscript.

Then change to the main Ghostscript directory of the MacTeX build system and type

```
sudo sh buildPackage.sh
```

This step will copy Ghostscript support files from /usr/local to the root directory of MacTeX's Ghostscript build folder. It will add the binaries constructed earlier, add "Fontmap" and "Fontmap.cmr" to root/usr/local/share/ghostscript/9.05/Resource/Init and add the

symbolic links to .pfb fonts to root/usr/local/share/ghostscript/9.05/Resource/Font. Finally, it will build the final root folder containing software to be installed, and make sure permissions are correct in this folder.

To finish, remove /usr/local and move /usr/local-temp back to /usr/local.

6. Adjusting Files in the Ghostscript Install Package

This package contains some files which are shown to the user during installation. One of these files is a brief Welcome document, a second is a longer ReadMe file, and a third shows the License for the code. After the Ghostscript install package is constructed, you may need to revise these documents. For example, the Welcome document contains the release date. Usually TeX Live authors are overly ambitious and predict a release several months earlier than it actually occurs, and you'll need to change the release date from time to time.

None of these files appear in MacTeX, but they appear if you make an optional Ghostscript Install Package.

7. Making a Standalone Install Package

This step is optional. If you want an install package containing only Ghostscript, construct it as follows.

This section of the MacTeX package contains a Packagemaker project file named "Ghostscript-Build". Edit this project file using the GUI interface of PackageBuilder as follows:

(1) Click the top item on the left. The right side will change to a view with three tabs. First select the Configuration tab. Edit so
   (a) Title: "Ghostscript-9.05'
   (b) User Sees: Easy Install Only
   (c) Install Destination: System volume
   (d) Certificate: Leave this blank. You might think that putting a certificate name here would correctly sign the package, but experiments show that it does not. The package needs to be signed after it is constructed.
   (e) Description: Empty

(2) The Requirements and Actions tab entries for the top item can be left blank

(3) Click the entry in the Contents section on the left. The right side will change to a view with two tabs. Select the Configuration tab. Edit so

   (a) Choice Name: Ghostscript-9.05

   (b) Identifier: choice1

   (c) Initial State: Selected and Enabled

   (d) Destiination: Leave Blank

   (e) Tooltip: Leave Blank

   (f) Description: Leave Blank

(4) The Requirements tab entries for this contents item can be left blank

(5) Open the Ghostscript-9.05 entry in the bottom left column. This item should show a "local" directory, obtained by dragging root/user/local to the left column. The right side will change to a view with four tabs. The item under the Components tab can be left blank. It is not necessary to edit the items under the Contents tab. The items under the Configuration tab should be edited as follows:

   (a) Install: root/usr/local (The item on the right should have selected "Relative To Project")

   (b) Destination: /usr/local (The item on the right should have selected "Relative To Project")

   (c) Do not check "Allow custom location"

   (d) Patch: No patch root selected

   (e) Package Identifier: org.tug.mactex.ghostscript9.05

   (f) Package Version: 1.0

   (g) Restart Action: None

   (h) Check "Require admin authentication"

   (i) PackageLocation: Self-Contained

   The items under the Scripts tab should be edited to read

   (a) Scripts directory: scripts (The item on the right should have selected "Relative To Project")

   (b) Preinstall: Leave Blank

   (c) Postinstall: scripts/postinstall (The item on the right should have selected "Relative To Project")

After verifying all these items, build the install package by clicking the "Build" icon. When asked to select a name, choose "Ghostscript-9.05-Temp". After the package has been made, sign it by executing the following line in Terminal:

```
sh signPackage.sh
```

This script simply calls

```
productsign --sign "Developer ID Installer: Richard Koch" old-name new-name
```

where "old-name" is Ghostscript-9.05-Temp.pkg and "new-name" is Ghostscript-9.05.pkg

Packagemaker will show a panel with a large number of warnings while building Ghostscript. Inspection of these warnings will confirm that all complain of permission errors in the root folder, and all complaints involve permissions assigned to symbolic links. I believe the current permissions are correct; they are certainly the same as permissions set by TeX Live when its binaries are created. So I ignore the warnings.