# DOC-TEXDIST

RICHARD KOCH

## 1. TeXLive-2021

This is the portion of the TUG documentation which deals with making the TeXDist package. This is used in the DVD version of the Installer, and can also be used to add the TeXDist structure to distributions installed on older machines by the Unix installer using the legacy binaries.

## 2. Updating for a New Year

To update the package for a new year, edit "buildPackage.sh" to reflect the new package names, which contain the year.

Also edit the License.rtf, ReadMe.rtf, and Welcome.rtf files both at the root level of the folder and inside the DVD folder.

Edit the two postflight scripts appropriately, mainly by changing the year. For instance, to change from 2020 to 2021, use Search and Replace to change the many, many occurrences of 2020 to 2021 in these files.

## 3. Creating the Package

Issue the command

```
sudo sh buildPackage.sh
```

to build the root folder, containing /Library/TeX.

## 4. Signing, Timestamping, and Adopting Hardened Runtimes for Binaries

This package has no binaries, so there is nothing to sign.

## 5. Building the Final Install Package

The install package is constructed using scripts in the folder "CreateInstallPackage." This folder also contains Resources and Scripts which will become part of the package. We first build TeXDist-2021Start.pkg by

```
sudo sh pkgbuildscript.sh
```

This script reaches up a level to find the root folder containing the package material. Next create TeXDist-2021-Temp.pkg by running a second script

```
sudo sh productbuildscript.sh
```

Finally, created a signed package TeXDist-2021.pkg by running the script

```
sh signPackage.sh
```

This script contains my Apple signing identity; future builders will first have to join Apple's Developer Support and get their own signing identity.

The CreateInstallPackage folder contains two other files and one other script. The files are requirement.xml and distribution.xml, and the script is create_distribution.sh. In future years, it may be enough to edit distribution.xml by hand. This file contains the TeXDist-2021Start.pkg name, and references to the background image, Welcome dialog, ReadMe dialog, and License dialog. It contains the identifier of the package, currently "org.tug.texdist2021". And it contains the restriction that the package can only be installed on machines running system 10.6 or higher.

But if necessary, the file can be recreated as follows. The system requirement is contained in requirement.xml. Edit this first. Then run the script create_distribution.sh. This script provides some information in distribution.xml, and outputs a sample distribution.xml file. But other items must be added by directly editing the file.

## 6. Notarizing the Final Install Package

Scripts to notarize TeXDist-2021.pkg are in the folder "NotarizePackage" inside "CreateInstallPackage." The actual TeXDist-2021.pkg created above can be left where it is; it need not be moved to the new folder.

To notarize,

```
sh sendnotarizerequest.sh
```

There will be a delay when nothing is printed in Terminal as the package is uploaded to Apple. When the upload is complete, a message of the form

```
RequestUUID = 96f0932f-b50f-4d73-ad65-1220fd9c8efe
```

will be printed. This magic number can be used to discover what went wrong if notarization fails, so I recommend copying it to the file requestUUID.tex. There will be a pause of about ten minutes and then a notification from Apple and an email from Apple will arrive. If all is well, this message says that notarization succeeded. To add the certificate to the install package

```
sh stapleresult.sh
```

The package TeXDist-2021.pkg is then ready for uploading and release.

In case of failure, open the script detailedinfo.sh and notice that the magic UUID number is part of this script. Copy and paste the new number from requestUUID.tex to this script. Then run it.

```
sh detailedinfo.sh
```

This command will return a web url. Use Safari to go to that url, where a very detailed error report will clearly state what went wrong.

The detailedinfo script runs a command named xcrun which communicates with developer support. Apple has adopted two-factor authentication security for such communication, but this security is not very convenient when running shell scripts. So developers can log into their Apple developer account and assign a password to xcrun and a few similar commands. Then when the script runs, typing this password is enough to run the command. Note that the detailedinfo script contains my account name and password. When someone else makes these packages, they will replace this information with their own personal information.