# TUG-BINARY

RICHARD KOCH

## 1. TeX Live Binaries

Starting in 2017, we support versions of macOS which are still receiving security updates from Apple. Thus in 2018 we should support the latest three systems, currently El Capitan, Sierra, and High Sierra. Shortly after TeX Live is released, Apple holds the developer conference, WWDC, and releases a first beta of the next system. This system is usually released in September. So for most of the year, we should support four versions of macOS.

In 2018, a very unusual event occurred. We compiled the binaries on El Capitan, and then discovered that they did not run on two machines running El Capitan, instead crashing immediately with an "undefined instruction" error. One of these machines was a MacBook introduced in 2007, and the other was an iMac introduced in 2007. Both of these machines were made obsolete by Sierra, which could not run on them.

Usually if we compile on a version of macOS, the binaries run on that version and all later versions. This is the first time binaries compiled on a version of macOS would not run on all machines running that same version of macOS. We can test three or four versions of macOS, but it would be impossible to test all Macintosh models running a particular version.

So we compiled the 2018 binaries on Yosemite; these binaries *did* run on the 2007 machines. This means that in 2018, we suppose four versions of macOS: Yosemite, El Capitan, Sierra, and High Sierra. In 2019, we will compile on Sierra and return to the policy of supporting the three most recent versions of macOS.

This is the portion of the TUG documentation which deals with making the TeX Live binaries. Building is a two stage process. First TeX Live and asymptote are compiled on the oldest supported system, currently Yosemite. Then the resulting binaries are processed here, forming the binaries sent to TUG for inclusion in TeX Live. (We sometimes build TeX Live on the latest operating system, currently High Sierra, to check that users who compile themselves will be happy.)

A clean package file will contain

- the folder AsymptoteBuild containing information for building asymptote.

*Date*: March 31, 2018.

- the folder BuildStatusDocs, containing technical details for building TeX Live and lisp
- the script "liposcript"
- the empty folder "RawCode"
- "TUG" folder with this document

The main MacTeX folder containing the Binary Folder must contain "xz", the binary which compresses packages, since the scripts in the Binary directory call it.

## 2. Compiling the Binaries

The "BuildStatusDocs" folder contains a crucial document named

`BuildStatus-2018`

This document contains the precise shell commands needed to obtain and build the TeX Live binaries. The document is an ordinary TextEdit plain text file so it can be easily edited if changes in the process are required. I find it convenient to put a copy of the document on the desktop of the build machine, merging editing changes back into the master document from time to time.

Building xindy requires obtaining and compiling CLISP. A separate document in the Build-StatusDocs folder explains how to do that.

Building asymptote requires obtaining, compiling, and installing three libraries. Complete instructions are included in the AsymptoteBuild folder, in the document

`AsymptoteBuild-2018`

The end result of these processes will be a folder of binaries and an isolated binary for asy. Place these items in the RawCode folder on this machine. Then run Terminal, change to the Binary directory, and run the command

```
sudo sh liposcript
```

This will create a folder named x86_64-darwin and a file named x86_64-darwin.tar.xz. Send the second xz file to Karl Berry.

## 3. Special Considerations for Xindy in 2018

We began compiling Xindy in 2010. In 2011, these binaries no longer worked. The author of Xindy pointed out that two lisp files did not change: xindy,mem and xindy.run. So from 2011 on, we compiled xindy using clisp, but replaced these two files with the 2010 versions.

In 2017, we adopted the modern approach of supporting only the three latest versions of macOS. That year's lisp files xindy.mem and xindy.run worked, so we used them rather than the 2010 versions. But this year, the 2018 versions of these files did not work, so at first we returned to the 2010 versions.

Herbert Schult pointed out to me that the 2010 version of xindy.run contains 32 bit binaries, but the 2017 version contains 64 bit binaries. Starting this year, Apple will begin phasing out 32 bit code on the Macintosh. So instead of using the 2010 version of the lisp files, this year and in the future we will use the 2017 versions of xindy.mem and xindy.run.

The liposcript which converts the binaries to x86_64-darwin automatically makes this substitution.

## 4. Special Considerations for LuaTeX in 2017 and 2018

In 2017, it was discovered that luaTeX could not call lua modules on the Macintosh. Ultimately this behavior was traced to the strip code. When TeX Live is built, part of the process calls "strip" to shrink the size of binaries by removing debugging code.

The Macintosh strip code has special flags which reduce the code size, but preserve luaTeX's ability to call lua modules. The required call, with flags, is "strip -u -r". Use of this call is somewhat controversial in the TeX Live community, but it turns out that the flags barely change code size over using the full "strip". So the code to build TeX Live 2018 is

```
cd
cd texlive2018dev/source
extern STRIP="strip -u -r"
./Build  --enable-xindy CLISP=/Users/koch/clisp/clisp-build/clisp
```

The Binary directory contains a folder named LuaNew with a small test of luaTeX to make sure that this fix works.