Setup OSX for Development

Bhaskar Karambelkar

Contents

Github API Token . . . 1
Install Homebrew and tap some taps . . . 1
Bash . . . 1
GCC Compiler and Autotools . . . 2
Misc libs . . . 3
SSL/SSH Libs . . . 4
GPG . . . 4
Java . . . 4
X-server and TCL-TK . . . 4
Python . . . 5
Git . . . 5
Boost libs w/ dependencies . . . 5
Latex Support . . . 6
R . . . 6
GIS Stuff . . . 8
Other Programming Languages . . . 8
Other Interesting Stuff . . . 9
GUI Apps . . . 9

Github API Token

Get a github account and setup an API token as described here.

```
echo 'export HOMEBREW_GITHUB_API_TOKEN="your_new_token"' >> $HOME/.bash_profile
. $HOME/.bash_profile
```

Install Homebrew and tap some taps

```
/usr/bin/ruby -e \
    "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
brew analytics off # full on paranoid mode
brew tap caskroom/cask # GUI apps
brew tap caskroom/fonts # fonts
brew tap homebrew/science
brew tap homebrew/completions
brew tap homebrew/services
brew tap homebrew/versions
```

Bash

Install Bash and verify

```
brew install bash bash-completion2
/usr/local/bin/bash --version
```

```
GNU bash, version 4.3.46(1)-release (x86_64-apple-darwin15.5.0)
```

Setup bash along with some utility functions.

```
sudo sh -c 'echo "/usr/local/bin/bash" >> /etc/shells'

mkdir -p $HOME/Library/Logs/Homebrew/$USER

echo '
if [ -f $(brew --prefix)/share/bash-completion/bash_completion ]; then
  . $(brew --prefix)/share/bash-completion/bash_completion
fi

brewPkg() {
  pkg=$1
  shift
  (
    brew install ${pkg} $*  2>&1 |
        tee $HOME/Library/Logs/Homebrew/$USER/${pkg}-$(date +"%F_%H%M").txt
  )
}

brewSrcPkg() {
  pkg=$1
  shift
  (
    brew install --build-from-source  ${pkg} $*  2>&1 |
        tee $HOME/Library/Logs/Homebrew/$USER/${pkg}-$(date +"%F_%H%M").txt
  )
}

brewSrcPkgWgcc() {
  pkg=$1
  shift
  (
    export CC=gcc-6
    export CXX=g++-6
    export HOMEBREW_CC=gcc-6
    export HOMEBREW_CXX=g++-6
    brew install --build-from-source  ${pkg} $*  2>&1 |
        tee $HOME/Library/Logs/Homebrew/$USER/${pkg}-$(date +"%F_%H%M").txt
  )
}

' >>  ~/.bash_profile
```

Load the new bash shell, so we can use all the auto-complete goodies. Or simply close and restart the Terminal App.

```
/usr/local/bin/bash -l
```

GCC Compiler and Autotools

Install gcc, without multilib so that OpenMP works.

```
brewPkg gcc --without-multilib
/usr/local/bin/gcc-6 --version
```

```
gcc-6 (Homebrew gcc 6.1.0 --without-multilib) 6.1.0
```

```
/usr/local/bin/gfortran --version
```

```
GNU Fortran (Homebrew gcc 6.1.0 --without-multilib) 6.1.0
```

Setup aliases for homebrew's gcc

```
cd /usr/local/bin
ln -s gcov-6 gcov
ln -s gcc-6 gcc
ln -s g++-6 g++
ln -s cpp-6 cpp
ln -s c++-6 c++
cd -
```

Install ccache to speed up compilation

```
brewPkg ccache
/usr/local/bin/ccache --version
```

```
ccache version 3.2.7
```

Install autotools, pkg-config, and cmake

```
brewPkg cmake pkg-config autoconf automake
```

Let's make sure OpenMP is working as expected.

```
cat > omp-test.c <<"END"
#include <omp.h>
#include <stdio.h>
int main() {
    #pragma omp parallel
    printf("Hello from thread %d, nthreads %d\n", omp_get_thread_num(), omp_get_num_threads());
}
END
gcc-6 -fopenmp -o omp-test omp-test.c
./omp-test
```

You should see something similar but not exactly the same.

```
Hello from thread 1, nthreads 8
Hello from thread 6, nthreads 8
Hello from thread 4, nthreads 8
Hello from thread 2, nthreads 8
Hello from thread 5, nthreads 8
Hello from thread 0, nthreads 8
Hello from thread 3, nthreads 8
Hello from thread 7, nthreads 8
```

Misc libs

```
brewPkg freetype fontconfig  pixman gettext
```

SSL/SSH Libs

Setup and verify openssl and libressl.

```
brewPkg  openssl
/usr/local/opt/openssl/bin/openssl version
```

```
OpenSSL 1.0.2h  3 May 2016
```

```
brewPkg  libressl
/usr/local/opt/libressl/bin/openssl version
```

```
LibreSSL 2.3.6
```

```
brewPkg libssh2
```

wget and curl

```
brewPkg wget
/usr/local/bin/wget --version
```

```
GNU Wget 1.18 built on darwin15.6.0.
```

```
brewPkg curl
/usr/local/opt/curl/bin/curl-config --version
```

```
libcurl 7.50.0
```

GPG

```
brewPkg gpg2 --with-readline
/usr/local/bin/gpg2 --version
```

```
gpg (GnuPG) 2.0.30
libgcrypt 1.7.2
```

```
brewPkg gpgme
/usr/local/bin/gpgme-config --version
```

```
1.6.0
```

Java

```
brew cask install java
java -version
```

```
java version "1.8.0_102"
Java(TM) SE Runtime Environment (build 1.8.0_102-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.102-b14, mixed mode)
```

X-server and TCL-TK

We need an X-Server. This takes a lot of time so be patient.

```
brew cask install xquartz
```

Python

Install python2

```
brewPkg python
pip install --upgrade pip setuptools
/usr/local/bin/python -V
```

```
Python 2.7.12
```

Install Python3

```
brewPkg python3
pip3 install --upgrade pip setuptools wheel
/usr/local/bin/python3 -V
```

```
Python 3.5.2
```

Git

```
brewPkg git --with-blk-sha1 --with-gettext \
    --with-pcre --with-persistent-https
/usr/local/bin/git --version
```

```
git version 2.9.2
```

Boost libs w/ dependencies

Install icu4c library for Unicode and globalization

```
brewPkg icu4c
/usr/local/opt/icu4c/bin/icu-config --version
```

```
57.1
```

Install libxml2, libiconv, and libxslt.

```
brewPkg libxml2 libiconv libxslt
/usr/local/bin/xml2-config --version
```

```
2.9.4
```

Install boost. Ignore the warning at the end.

```
brewPkg boost --with-icu4c --with-mpi --without-single
```

Boost is a beast of a library, so we need some quick programs to test whether it has installed successfully. Shamelessly copied/adapted from the Intertubes. Ignore the warning messages spewed by the compiler.

```
cat > first.cpp <<END
#include<iostream>
#include<boost/any.hpp>
int main()
{
    boost::any a(5);
    a = 1.61803;
    std::cout << boost::any_cast<double>(a) << std::endl;
}
```

```
END
clang++ -o first first.cpp
./first
```

```
1.61803
```

```
cat > second.cpp <<END
#include<iostream>
#include <boost/filesystem.hpp>
int main()
{
    boost::filesystem::path full_path( boost::filesystem::current_path() );
    if ( boost::filesystem::exists( "second.cpp" ) )
    {
        std::cout << "Found second.cpp file in " << full_path << std::endl;
    } else {
        std::cerr << "Argh!, Something not working" << std::endl;
        return 1;
    }
}
END
clang++ -o second second.cpp \
    -lboost_filesystem-mt -lboost_system-mt
./second
```

```
Found second.cpp file in "/Users/brewmaster"
```

Latex Support

Take a Coffee break and then some, because this is a huge one.

```
brew cask install mactex
```

R

Install libsvg, librsvg, and cairo.

```
brewPkg cairo
brewPkg libsvg
brewPkg librsvg
brewPkg pandoc
```

Openblas for speedier linear algebra in R.

```
brewPkg openblas --with-openmp
```

Test openblas. Shamelessly copied from Intertubes.

```
cat > test-openblas.c <<"END"
#include <cblas.h>
#include <stdio.h>

void main()
{
  int i=0;
  double A[6] = {1.0,2.0,1.0,-3.0,4.0,-1.0};
```

```
  double B[6] = {1.0,2.0,1.0,-3.0,4.0,-1.0};
  double C[9] = {.5,.5,.5,.5,.5,.5,.5,.5,.5};
  cblas_dgemm(CblasColMajor, CblasNoTrans, CblasTrans,
    3,3,2,1,A, 3, B, 3,2,C,3);

  for(i=0; i<9; i++)
    printf("%lf ", C[i]);
  printf("\n");
}
END

clang -L/usr/local/opt/openblas/lib \
    -I/usr/local/opt/openblas/include \
    -lopenblas -lpthread \
    -o test-openblas test-openblas.c

./test-openblas
```

```
11.000000 -9.000000 5.000000 -9.000000 21.000000 -1.000000 5.000000 -1.000000 3.000000
```

Eigen and Armadillo for Rcpp and v8 for R+v8.

```
brewPkg eigen
brewPkg armadillo --with-hdf5
brewPkg v8 --with-icu4c --with-readline
```

Test Armadillo

```
cd /usr/local/opt/armadillo/examples/
clang++   -O2   -o example1  example1.cpp  -larmadillo -framework Accelerate
./example1
```

You should see something like below and lot more.

```
Armadillo version: 7.200.2 (Plutocratic Climate Change Denialist)
A.n_rows: 2
A.n_cols: 3
...
```

Test v8

```
echo 'quit()' | v8
```

```
V8 version 5.1.281.47 [sample shell]
```

Finally install R itself, and also setup R to use Apple's `clang` compiler. We can also setup R to use `gcc` compiler to take advantage of `openmp` support, but I've noticed that not all R packages compile correctly when using GCC.

```
brewPkg r --with-openblas --with-pango
# for rJava to work.
R CMD javareconf \
    JAVA_CPPFLAGS=-I/System/Library/Frameworks/JavaVM.framework/Headers

# Setup $HOME/.r/Makevars file to properly link against homebrew packages.
mkdir $HOME/.r
cat > $HOME/.r/Makevars << END
CC=ccache clang
```

```
CXX=ccache clang++
SHLIB_CXXLD=ccache clang++
FC=gfortran-6
F77=gfortran-6
MAKE=make -j8
PKG_CONFIG_PATH=/usr/local/lib/pkgconfig:/opt/X11/lib/pkgconfig
END

# Setup $HOME/.r/Makevars file to properly load against homebrew libs.
# Also setup a R_LIBS_USER directory to install R packages locally.
mkdir -p $HOME/Library/R/3.3/library
cat > $HOME/.Renviron <<END
export R_LIBS_USER=$HOME/Library/R/3.3/library
END

# add same stuff to .bash_profile
cat $HOME/.Renviron >> $HOME/.bash_profile
```

GIS Stuff

Mostly PostGIS + Geo libs.

```
brewPkg postgresql
brewPkg geos
brewPkg gdal \
    --with-armadillo --with-complete --with-libkml \
    --with-opencl --with-postgresql --with-unsupported
brewPkg postgis --with-gui
```

Other Programming Languages

Some other programming languages I use occasionally.

Node.js

```
brewPkg node
brewPkg phantomjs casperjs
```

Scala

```
brew install scala
```

golang

```
brew install golang
cat >> $HOME/.bash_profile <<END
export GOPATH=$HOME/golang
export GOROOT=/usr/local/opt/go/libexec
export PATH=$PATH:$GOPATH/bin
export PATH=$PATH:$GOROOT/bin
END
```

Other Interesting Stuff

```
brewPkg imagemagick --with-fontconfig --with-ghostscript \
    --with-librsvg --with-openmp --with-pango --with-webp
brewPkg vim --with-python3 --with-client-server \
    --with-lua --with-luajit --with-override-system-vi

brewPkg macvim --with-python3 --with-client-server \
    --with-lua --with-luajit --with-override-system-vim

brewPkg jq # json processing

brew cask install font-hack font-fira-code #extra fonts

brewPkg mlpack # Fast Machine Learning
brewPkg protobuf --devel # Google's Protocol Buffer Library
brewPkg gsl # GNU Scientific Library
brewPkg libyaml # YAML Support
```

GUI Apps

Apps related to Securing your Mac.

```
brew cask install blockblock knockknock \
    dhs taskexplorer kextviewr
brew cask install suspicious-package
```

Quicklook Plugins for Developers.

```
brew cask install qlcolorcode qlstephen qlmarkdown \
    quicklook-json qlprettypatch quicklook-csv betterzipql \
    qlimagesize webpquicklook
```

These are some GUI apps I use, pick and chose as you like.

```
brew cask install google-chrome chrome-devtools firefox iterm2 seil \
 slate keepassx free-mind itsycal flux caffeine alfred beardedspice \
 macdown mysqlworkbench osxfuse smcfancontrol torbrowser vagrant\
 vagrant-manager vlc cog yed slack owncloud
```