Local Development Setup

Table of Contents

Operating System

macOS

- Latest version of macOS for development
- Set up full disk encryption, enable firewall
- Create a standard user account for daily use: first user is always an admin
- Install apps with Homebrew Cask under `~/Applications`

Installations

Over time I have found out that installations under a user's home directory is less problematic and tend to cause less permission errors. For that reason, I will be installing everything under `~` as long as it's possible.

Package Managers

Homebrew

Install Xcode Command Line Tools:

```
xcode-select --install
```

Install Homebrew:

```
mkdir .homebrew && curl -L https://github.com/Homebrew/brew/tarball/master | tar xz --strip 1 -C .homebrew
```

Add `~/.homebrew/bin` and `~/.homebrew/sbin` to your `PATH` in `~/.zshrc`:

Example 1. Adding homebrew to your path

```
function path {
  if [[ -d "$1" ]] ; then
    if [[ -z "$PATH" ]] ; then
      export PATH=$1
    else
      export PATH=$PATH:$1
    fi
  fi
}

export PATH=''
path ~/.homebrew/sbin ①
path ~/.homebrew/bin ②
path /usr/local/sbin
path /usr/local/bin
path /usr/sbin
path /usr/bin
path /sbin
path /bin
```

Check if everything is working properly:

```
brew doctor
```

You will see the following warning, which is expected:

> Warning: Your Homebrew’s prefix is not /usr/local. Some of Homebrew’s bottles (binary packages) can only be used with the default prefix (/usr/local).

Software

Brewfile

You can use Homebrew Bundle to systematically install software:

```
brew bundle  # --file=~/Brewfile
```

For this to work, you need to create a Brewfile. Check `config/Brewfile` for an example.

Apart from the initial installation, it is also possible to enforce your Brewfile:

```
brew bundle cleanup --force  # --file=~/Brewfile
```

This will remove any package that’s not present or dependent to a package listed in your Brewfile.

Z Shell

Install ZSH using Homebrew:

```
brew install zsh
```

Install Zinit to manage plugins:

```
mkdir ~/.zinit
git clone https://github.com/zdharma/zinit.git ~/.zinit/bin
```

Get ZSH config files from my dotfiles repo:

```
curl -L https://git.io/fjgjN > ~/.zshrc
```

Switch non-admin user’s shell to ZSH:

```
su - admin
sudo dscl . -create /Users/kerem UserShell /Users/kerem/.homebrew/bin/zsh ①
```

① Replace `kerem` with your user

Version Managers

Ruby Version Manager

Import GPG Keys:

```
gpg --keyserver hkp://keys.gnupg.net \
    --recv-keys 409B6B1796C275462A1703113804BB82D39DC0E3 7D2BAF1CF37B13E2069D6956105BD0E739499BDB
```

Download the Installer:

```
\curl -O https://raw.githubusercontent.com/rvm/rvm/master/binscripts/rvm-installer
\curl -O https://raw.githubusercontent.com/rvm/rvm/master/binscripts/rvm-installer.asc
```

Verify Installer Signature:

```
gpg --verify rvm-installer.asc
```

Run the Installer:

```
bash rvm-installer --branch stable --ignore-dotfiles
```

Load RVM into shell session (update `.zshrc`):

```
source ~/.rvm/scripts/rvm
```

Remove artifacts:

```
rm rvm-installer
rm rvm-installer.asc
```