RITS Training Installation Instructions

University College London

1 Installation Instructions

1.1 Introduction

This document contains instructions for installation of the packages we'll be using during the UCL Software Carpentry Workshops and our other training courses. You will be following the training on your own machines, so please complete these instructions.

1.2 What we're installing

For the software carpentry session on programming, we'll be using the language *Python* and in particular version 2.7. We will use the *Enthought Canopy* python distribution which contains a good collection of the most common *Python* modules as well as *IPython* (an improved *Python* interpreter) and the IPython notebook (a useful web-based user interface that allows you to create documents that combine text and *Python* code, executable with the same browser window). We'll need *pip*, the package installer for *Python*, as well.

For the workshop session on version control, we'll be using *Git* and the *Github* website.

If you're just attending the software carpentry course, and not other RITS courses, you can ignore the sections on compilers, cmake, and subversion. However, we encourage you to be ready to follow our other courses, so you might want to complete these instructions too.

1.3 A note on systems

Linux users should be able to use their package manager to install all of this software (if you're using Linux, we assume you won't have any trouble with these requirements).

However note that if you are running an older Linux distribution you may get older versions with different look and features. A recent Linux distribution is recommended.

Mac and Windows users should follow the instructions below.

1.4 Don't panic

Give yourself 30 minutes or so to run through this installation process and don't get intimidated! Please try to install everything well before the bootcamp, as it is important that we don't waste time during the workshop trying to mend installations.

If you do get stuck, first try searching on the internet (e.g. stackoverflow.com[1]) for solutions. Or, try asking a fellow bootcamp attendee for help.

1.5 Drop-in session

We will be running a drop-in session for people that would like some help with setting up their laptop in advance of the training. You will have been advised as to when this will occur.

All students should either attend this or ensure that they have a working git, python, and editor and shell installation by following the instructions bellow before the workshop. Eduroam ——-

We will be using UCL's eduroam[2] service to connect to the internet for this work.

So you should ensure you have eduroam installed and working.

2 Linux

2.1 Python via package manager

Recent versions of Ubuntu (12.10 and 13.04) pack mostly up to date versions of all needed packages. You should ensure that the following packages are installed:

[1] http://stackoverflow.com
[2] http://www.ucl.ac.uk/isd/staff/wireless/eduroam

- python-numpy
- python-scipy
- python-nose
- python-matplotlib
- python-pip
- ipython
- ipython-notebook

Older distributions may have outdated versions of specific packages. Other linux distributions most likely also contain the needed python packages but again they may also be outdated.

2.2 Python via Enthought Canopy

Alternatively you may install a complete independent scientific python distribution. One of these is Enthought Canopy.

The Enthought Canopy python distribution exists in two different versions. A basic free version with a limited number of packages (Canopy express) and a non free full version. The full version can be obtained free of charge for academic use. Register with Enthought Scientific Computing[3] using your UCL e-mail address for an academic licence.

You may then use your Enthought user account to sign into the installed Canopy application and activate the full academic version. Canopy comes with a package manager from where it is possible to install and update a large number of python packages. The packages installed by default should cover our needs.

2.3 Git

If git is not already available on your machine you can try to install it via your distribution package manager (e.g. `apt-get` or `yum`), for example:

```
sudo apt-get install git
```

2.4 Editor

Many different text editors suitable for programming are available. If you don't already have a favourite, you could look at Kate[4].

Regardless of which editor you have chosen you should configure git to use it. Executing something like this in a terminal should work:

[3] https://enthought.com/products/canopy/academic/
[4] http://kate-editor.org/

```
git config --global core.editor NameofYourEditorHere
```

The default shell is usually bash but if not you can get to bash by opening a terminal and typing `bash`.

2.5 Compilers

Those considering following other RITS courses beyond the should also complete the Compiler setup instructions[5].

2.6 Subversion

Again, install the appropriate package with apt-get or yum, for example:

```
sudo apt-get install subversion
```

2.7 CMake

Again, install the appropriate package with apt-get or yum (`cmake`)

3 Mac

3.1 Python

The Enthought Canopy python distribution exists in two different versions. A basic free version with a limited number of packages (Canopy express) and a non free full version. The full version can be obtained free of charge for academic use. Register with Enthought Scientific Computing[6] using your UCL e-mail address for an academic licence.

You may then use your Enthought user account to sign into the installed application and activate the academic version. Canopy comes with a package manager from where it is possible to install and update a large number of python packaged. The packages installed by default should cover our needs.

If you use this route, you can ignore the distribute & pip section.

[5] http://development.rc.ucl.ac.uk/training/CPP/#/post-carpentry-installation
[6] https://enthought.com/products/canopy/academic/

3.2 Python from python.org

Alternative, you can download *Python* from python.org:

If you have a newer Mac, i.e. one from the last few years, you should download this version[7] and follow the instructions[8] about Tcl/Tk dependencies.

If you have an older Mac, follow these instructions[9], including OS-version-specific information and important details about Tcl/Tk/TKinter dependencies.

3.3 Distribute and Pip

Open terminal (search in spotlight for "terminal" or look in the Applications/Utilities folder) and paste in this:

```
curl -O http://python-distribute.org/distribute_setup.py
python distribute_setup.py
sudo easy_install pip
```

You will need to enter your administrator password.

3.4 XCode and Command line tools

Install XCode[10] using the Mac app store.

We do not recommend following this training on older versions of OSX without an app store: upgrade to OSX Mavericks.

Then, go to Xcode…Preferences…Downloads… and install the command line tools option

3.5 Git

Install Homebrew via typing this at a terminal:

```
ruby -e "$(curl -fsSL https://raw.github.com/mxcl/homebrew/go)"
```

and then type

```
brew install git
```

Then install the GitHub for Mac client[11].

[7] http://www.python.org/ftp/python/2.7.5/python-2.7.5-macosx10.6.dmg
[8] http://www.python.org/download/mac/tcltk/
[9] http://www.python.org/getit/releases/2.7.5/
[10] https://itunes.apple.com/us/app/xcode/id497799835
[11] http://mac.github.com

3.6 Git Without Homebrew

Alternatively, you can install Git directly without Homebrew via Git[12]. Ignore the 'set up git' section. Homebrew is the better route, as it will then be easier to obtain other tools you may need later.

3.7 CMake

Just do

```
brew install cmake
```

(If you're not using homebrew, you can get the package directly[13])

3.8 Editor and shell

The default text editor on OS X *textedit* should be sufficient for our use. Alternatively http://mac.appstorm.net/roundups/office-roundups/top-10-mac-text-editors/ lists a number of other good editors.

To setup git to use *textedit* executing the following in a terminal should do.

```
git config --global core.editor /Applications/TextEdit.app/Contents/MacOS/
```

The default terminal on OSX should also be sufficient. If you want a more advanced terminal iTerm2[14] is an alternative.

4 Windows

4.1 Python

The Enthought Canopy python distribution exists in two different versions. A basic free version with a limited number of packages (Canopy express) and a non free full version. The full version can be obtained free of charge for academic use. Register with Enthought Scientific Computing[15] using your UCL e-mail address for an academic licence. After registering download and install the Canopy package for your operating system.

You may then use your Enthought user account to sign into the installed application. Canopy comes with a package manager from where it is possible to install and update all the shiped python packaged. The packages installed by default should cover our needs.

[12] https://help.github.com/articles/set-up-git
[13] http://www.cmake.org/cmake/resources/software.html
[14] http://www.iterm2.com/
[15] https://enthought.com/products/canopy/academic/

4.2 Sophos

To use the IPython notebook on a Windows computer with Sophos antivirus installed it may be necessary to open additional ports allowing communication between the notebook and its server. The solution[16] is:

- open your Sophos Endpoint Security and Control Panel from your tray or start menu
- Select "configure" > "Anti-virus" > "Authorization" from the menu at the top
- Select the websites tab
- click the "Add" button and add 127.0.0.1 and localhost to the "Authorized websites" list
- restart computer (most likely not needed, just restart the IPython notebook)
- output works now :)

4.3 Git

Install msysgit[17]

Then install the GitHub for Windows client[18].

4.4 Editor

Unless you already use a specific editor which you are comfortable with we recommend using Notepad++[19]. Follow the download link.[20]

Using Notepad++ to edit text files including code should be straight forward but in addition you should configure git to use notepad++ when writing commit messages (We will learn about these in the version controle session).

4.5 The Shell

Install MinGW[21] by following the download link. It should install MinGW's package manager. On the left, select `Basic Setup`, and on right select `mingw32-base`, `mingw-developer-toolkit`, `mingw-gcc-g++`, `msys-base`. On some systems these package might be selected from start. Finally, click the installation menu and `Apply Changes`.

[16] http://stackoverflow.com/questions/13036197/ipython-notebook-getting-output
[17] http://code.google.com/p/msysgit/downloads/list?q=full+installer+official+git
[18] http://windows.github.com/
[19] http://notepad-plus-plus.org/
[20] http://notepad-plus-plus.org/
[21] http://sourceforge.net/projects/mingw/

4.6 Finding out where things got installed

Now, we need to find out where Git and Notepad++ have been installed, this will be either in `C:/Program Files (x86)` or in `C:\ProgramFiles`. The former is the norm on more modern versions of windows. If you have the older version, replace `Program\ Files\ \(x86\)` with `Program\ Files` in the instructions below.

4.7 Telling the shell where to find Notepad++

We need to tell the new shell installed in this way where git and Notepad++ are.

To do this, use NotePad++ to edit the file at `C:\MinGW\mysys\1.0\etc\profile`

and toward the end, above the line `alias clear=clsb` add the following:

```
# Path settings from SoftwareCarpentry
export PATH=$PATH:/c/Program\ Files\ \(x86\)/Git/bin
export PATH=$PATH:/c/Program\ Files\ \(x86\)/Notepad++
# End of Software carpentry settings
```

4.8 Testing your install

Check this works by opening MinGW shell, with the start menu (Start->All programs->MinGW->MinGW Shell). This should open a *terminal* window, where commands can be typed in directly. On windows 8, there may be no app for MinGW. In that case, open the `run` app and type in `C:\MinGW\msys\1.0\msys.bat`. Once you have a terminal open, type

```
which notepad++
```

which should produce readout similar to `/c/Program Files (x86)/Notepad++/notepad++.exe`

```
which git
```

which should produce `/c/Program Files (x86)/Notepad++/notepad++.exe`. The `which` command is used to figure out where a given program is located on disk.

4.9 Telling Git about Notepad

Now we need to update the default editor used by Git.

```
git config --global core.editor "'C:/Program Files (x86)/Notepad++/notepad+
```

Note that it is not obvious how to copy and paste text in a Windows terminal including Git Bash. Copy and paste can be found by right clicking on the top bar of the window and selecting the commands from the drop down menu (in a sub menu).

4.10 Testing python

Confirm that the Python installation has worked by typing:

```
python -V
```

Which should result in details of your installed python version.

This should print the installed version of the python and git confirming that both are installed at working correctly.

You should now have a working version of git, python, and notepad++, all accessible from your shell.

4.11 Subversion

Install subversion[22]

And choose to add it to the path for all users if so prompted.

4.12 CMake

Install cmake[23]

And choose to add it to the path for all users if so prompted.

4.13 Testing cmake and subversion

Check that cmake and subversion work from your MSys shell:

```
which cmake
which svn
```

[22] http://sourceforge.net/projects/win32svn/
[23] http://www.cmake.org/cmake/resources/software.html