Setting up a PyQGIS development machine

GEO1005 (2017-18 Q2)

Spatial Decision Support for Planning and Crisis Management

Jorge Gil

Developing Python plugins for QGIS requires the installation and configuration of several software packages.

It is a complicated process, but once completed, making plugins is a smooth procedure and you can focus on implementing the functionality and testing the interaction of your plugin.

After completing these steps, your machine will be ready to make sophisticated plugins.

1. Install QGIS 2.18 LTR
2. Install ‘Plugin Builder’ and ‘Plugin Reloader’ QGIS plugins
3. Install ‘pip’
4. Install the ‘Plugin Builder Tool’ (pb_tool)
5. Mac/Linux: Install Qt Creator
6. Mac/Linux: Setup the environment paths
7. Install PyCharm
8. Windows: Setup PyCharm shortcut
9. Configure PyCharm

Check if OSGeo4W shell is installed correctly:

- Locate the QGIS shortcut
- Locate the Qt Designer / Creator shortcut
- Find OSGeo4W shell shortcut
- Run shell as administrator
- Type ‘pyrcc4’ and press enter
- If it gives an error move to the next step
- Otherwise jump to slide 6

Only if the previous step fails!

- Run the QGIS OSGeo4W Network installer
- Choose *Advanced Install* and select to install:
  - Libs -> qt4-devel
  - Libs -> python-pip
  - Libs -> setuptools
- If not present download the QGIS OSGeo4W Network installer version and install it:

http://www.qgis.org/en/site/forusers/download.html

- Choose to install Desktop -> qgis-ltr

http://www.kyngchaos.com/software/qgis

"The beast is actively interested only in now, and, as it is always now and always shall be, there is an eternity of time for the accomplishment of objects."
- the wisdom of Tarzan

KyngChaos Wiki

Log In

You are here: KyngChaos Wiki » Software » QGIS

Main Menu

Anime & Manga
Mac OS X Porting
Software

Software Menu

SumomOS
UNIX Porting Downloads

- Frameworks
- QGIS
- PostgreSQL
- Python Modules
- GRASS GIS
- PHP
- MapServer
- Download Archive

FAQ
Installation Guide
Developer Notes

Search Search

QGIS

Table of Contents
- QGIS
  - Current
  - Long Term Support
  - Development Builds

Mac OS X installers for QGIS. For OS X Mt Lion and newer. Install the Current version to stay up to date on features. Install the Long Term Support version for feature stability for a year.

Install Note: OS X security may block installation, as I'm not an "identified developer" (ie not paying Apple to develop software). There a simple way to force it to install - right-click the installer file and select **Open**, this will trigger an extra option in the security warning to install it anyways. You only have to do this once, the system will remember the QGIS installer for any future installs (new major versions *may* reset this).

Current

All required items are included on the disk image.

WARNING: QGIS *will* crash if Qt4 developer components are installed in the standard /Developer location. Either rename /Developer/Applications/Qt/Plugins or uninstall Qt before running QGIS.

Optional:

- other Python Modules for plugins

Download:

- QGIS 2.18.0-1 [286.4 MiB].

See the QGIS website for more information about, and help with, QGIS.

QGIS includes its own internal copies of GRASS, Orfeo Toolbox, SAGA and TauDEM.

Long Term Support

All required items are included on the disk image.

WARNING: QGIS *will* crash if Qt4 developer components are installed in the standard /Developer location. Either rename /Developer/Applications/Qt/Plugins or uninstall Qt before running QGIS.

Optional:

- other Python Modules for plugins

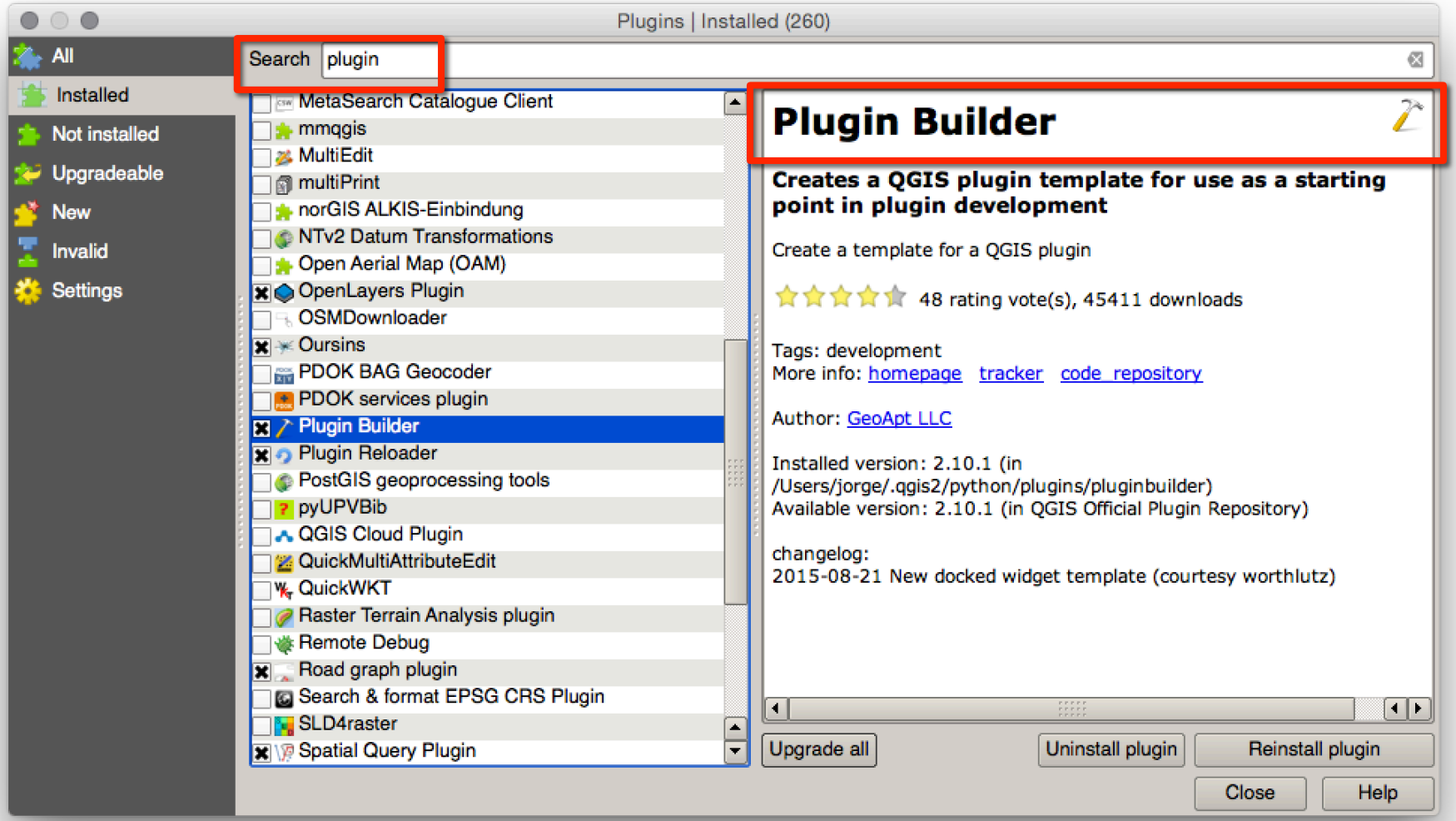
Download:

- QGIS 2.14.8-1 [270.1 MiB].

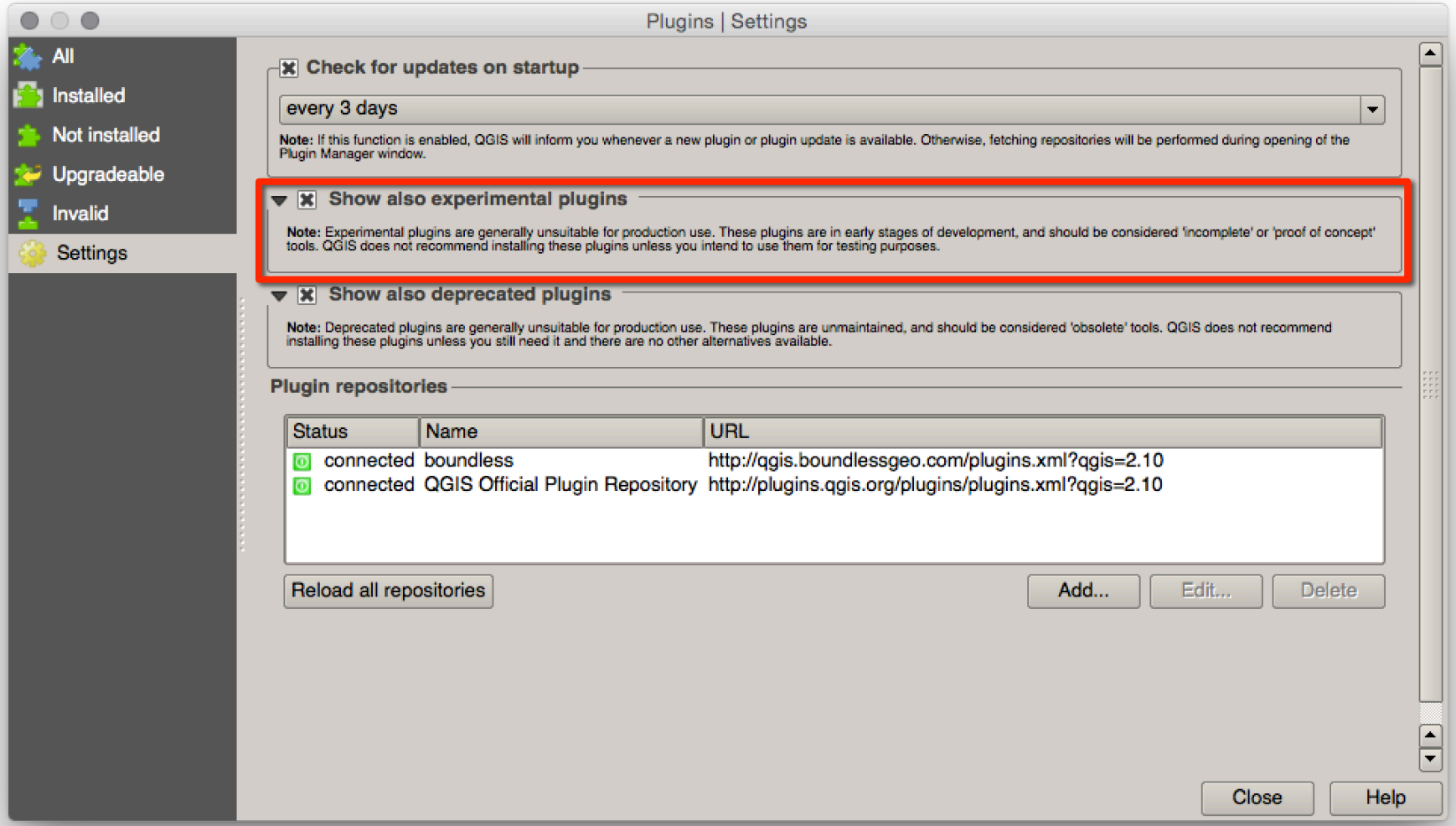
See the QGIS website for more information about, and help with, QGIS.

QGIS includes its own internal copies of GRASS 6, Orfeo Toolbox, SAGA and TauDEM.

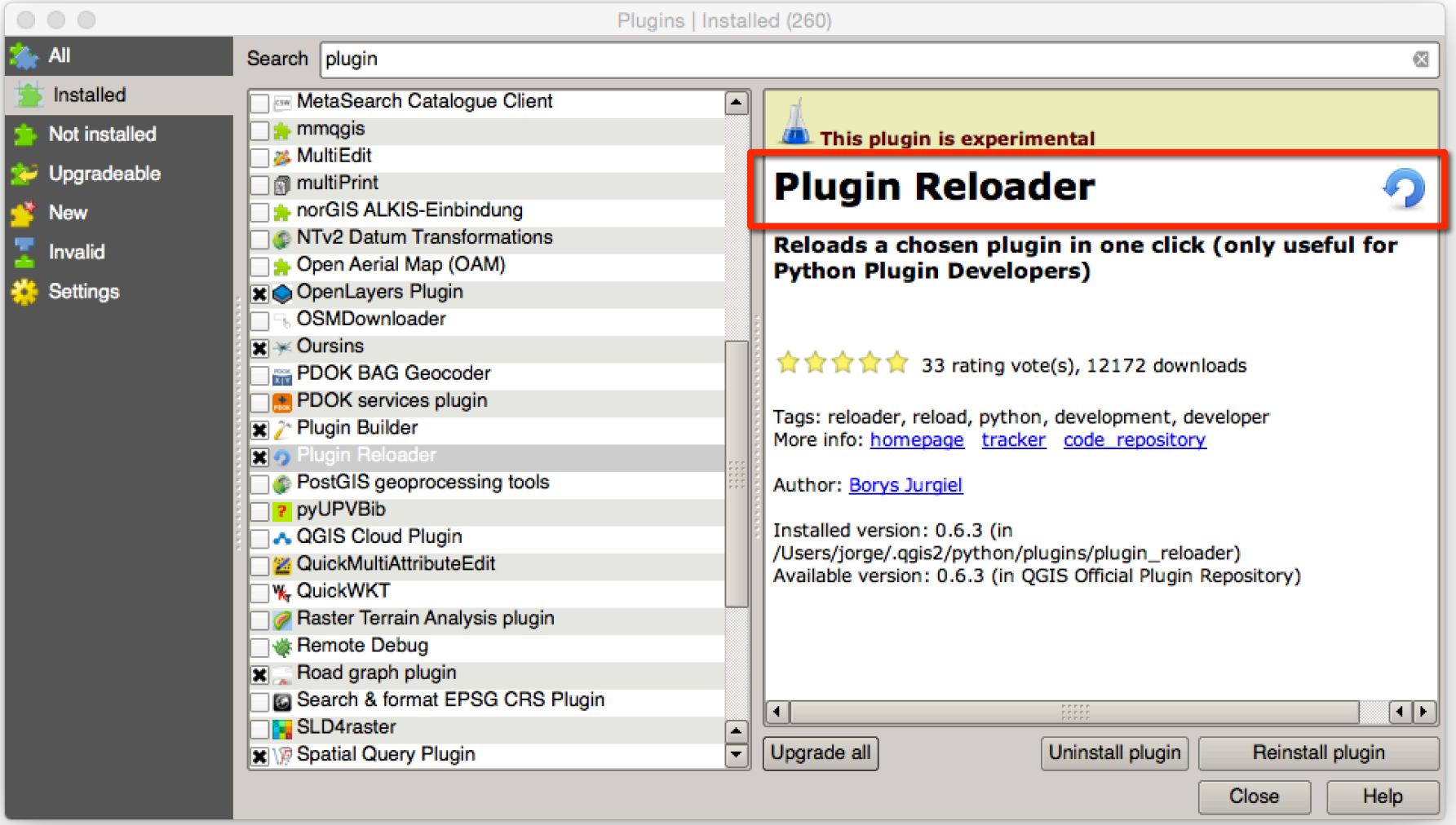
Start QGIS and go to Plugins Manager. Use the search to easily find the plugins you need.



In Settings Check the box to display experimental plugins.



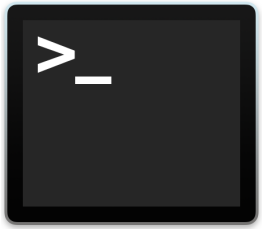
In Settings Check the box to display experimental plugins.



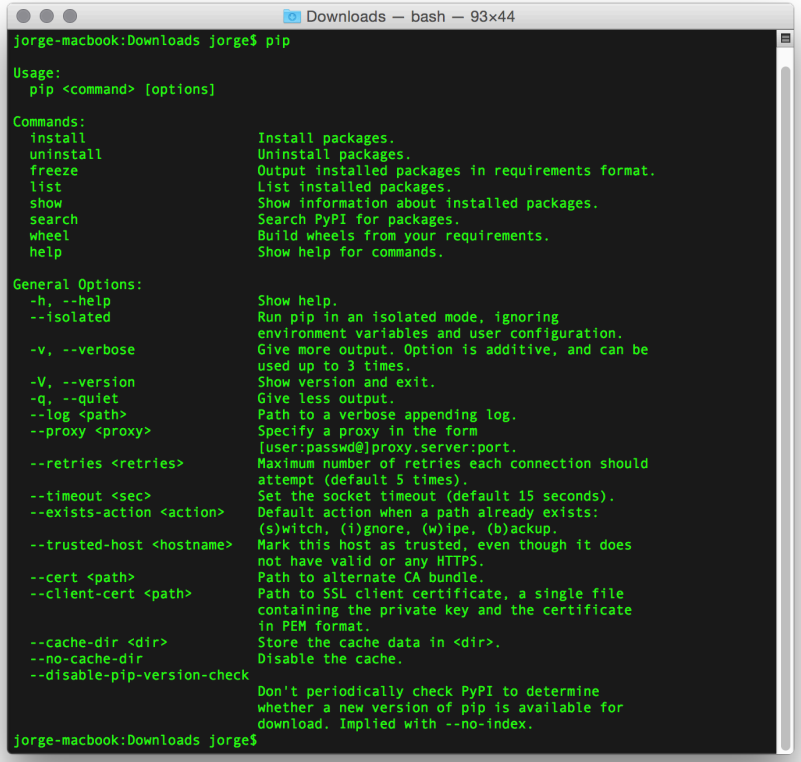
Pip is a command line Python package installer.

Check if you have pip:

1. Windows: start the OSGeo4W Shell command line window as administrator;
2. Mac OSX: start the terminal;



3. Type ‘pip’ on the command prompt;
4. Press Enter.
5. If you see a screen ason the right… pip is already installed,
   skip the next step!

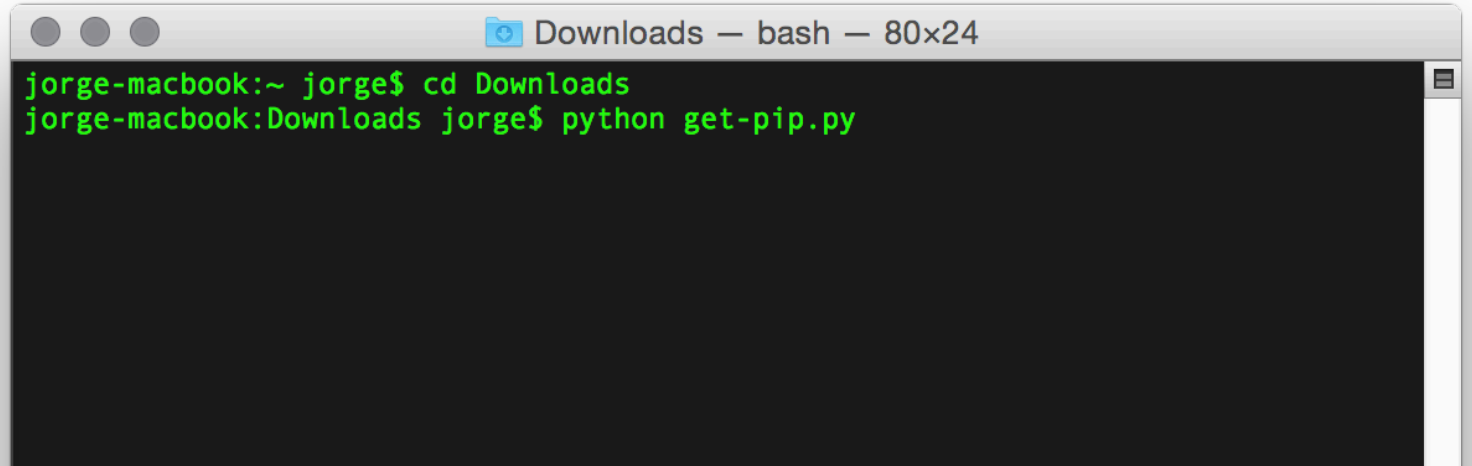


It is available from the OSGeo4W network installer, under libs (python-pip).

Otherwise download: https://bootstrap.pypa.io/get-pip.py

Install ‘pip’:

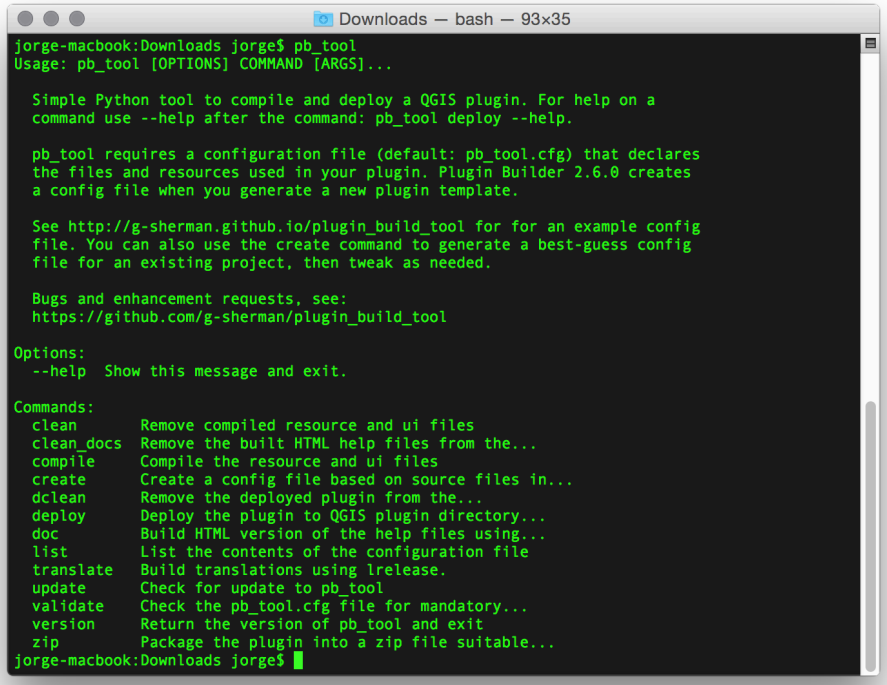
1. Using the Shell or Terminal, change directory to the download location (cd [path])
2. Type and execute:‘python get-pip.py
3. MacOSX: sudo python get-pip.py
4. Wait for the installation to complete and it should work…
5. Test by typing and executing: pip



Install the ‘Plugin Builder Tool’ using the command line

To install use the Shell or the Terminal:

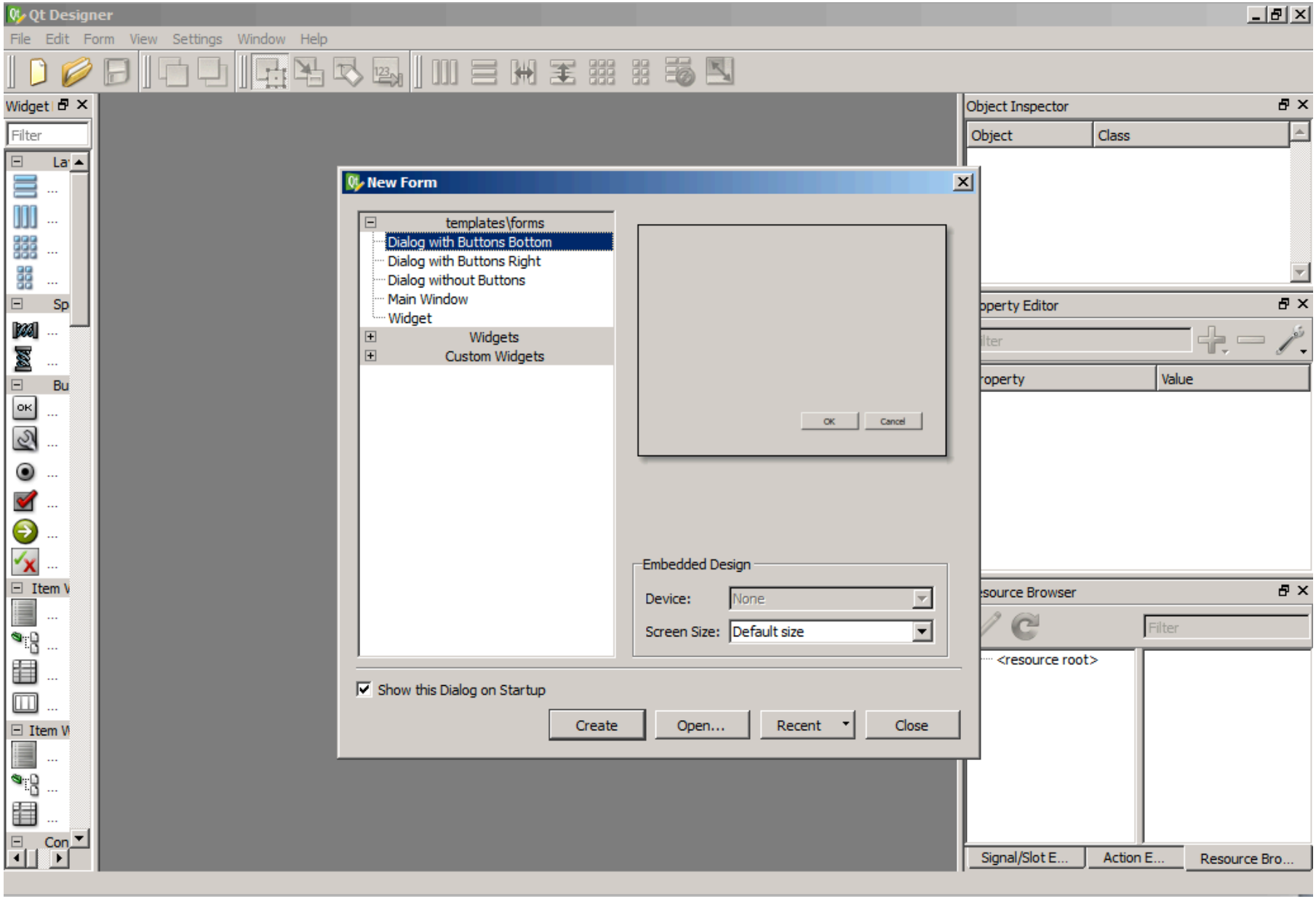
1. Type and execute ‘pip install pb_tool’
2. On MacOSX: ‘sudo pip install pb_tool’
3. Close and reopen the terminal window
4. Type and execute ‘pb_tool’ to test
5. You should see the screen on the right…



For details see the website:

https://g-sherman.github.io/plugin_build_tool/

Locate Qt Designer / Creator in the start menu



(Mac OSX only) Install developer tools and QtCreator

- Install Mac OSX developer tools from the ‘App Store’
- Start the developer tools to agree with the license terms
- Close Xcode
- Download and install Qt Creator:

https://download.qt.io/official_releases/qtcreator/4.4/4.4.1/qt-creator-opensource-mac-x86_64-4.4.1.dmg

(Mac OSX only) Configure environment paths

Edit the .bash_profile in your Home folder to use the QGIS app folder and the included Python packages.

Start the Terminal, and type:

1. to create the file if it doesn't exist:
   - **`touch ~/.bash_profile`**
2. to open file for editing:
   - **`open ~/.bash_profile`**

Write the following commands (in bold) in the text file:

1. **`QGISBASE="/Applications/QGIS.app/Contents"`**
2. `export PATH=/usr/bin:/bin:/usr/sbin:/sbin:/usr/local/bin:/System/Library/Frameworks/Python.framework/Versions/2.7/bin/`**`:${QGISBASE}/MacOS/bin`**
3. **`export DYLD_LIBRARY_PATH=${QGISBASE}/MacOS/lib`**
4. **`export PYTHONPATH=:${QGISBASE}/Resources/python:$PYTHONPATH`**

Ready to develop QGIS plug-ins!

(but it can be better…)

An IDE (Integrated Development Environment) does much more than a simple code editor:

- create projects and manage related files
- refactor modules and variable names
- syntax highlighting
- code completion
- checks code for compliance with standards
- runs compilation and other tools
- synchronises with the github repository
- has debugging tools

PyCharm is a popular Python IDE.

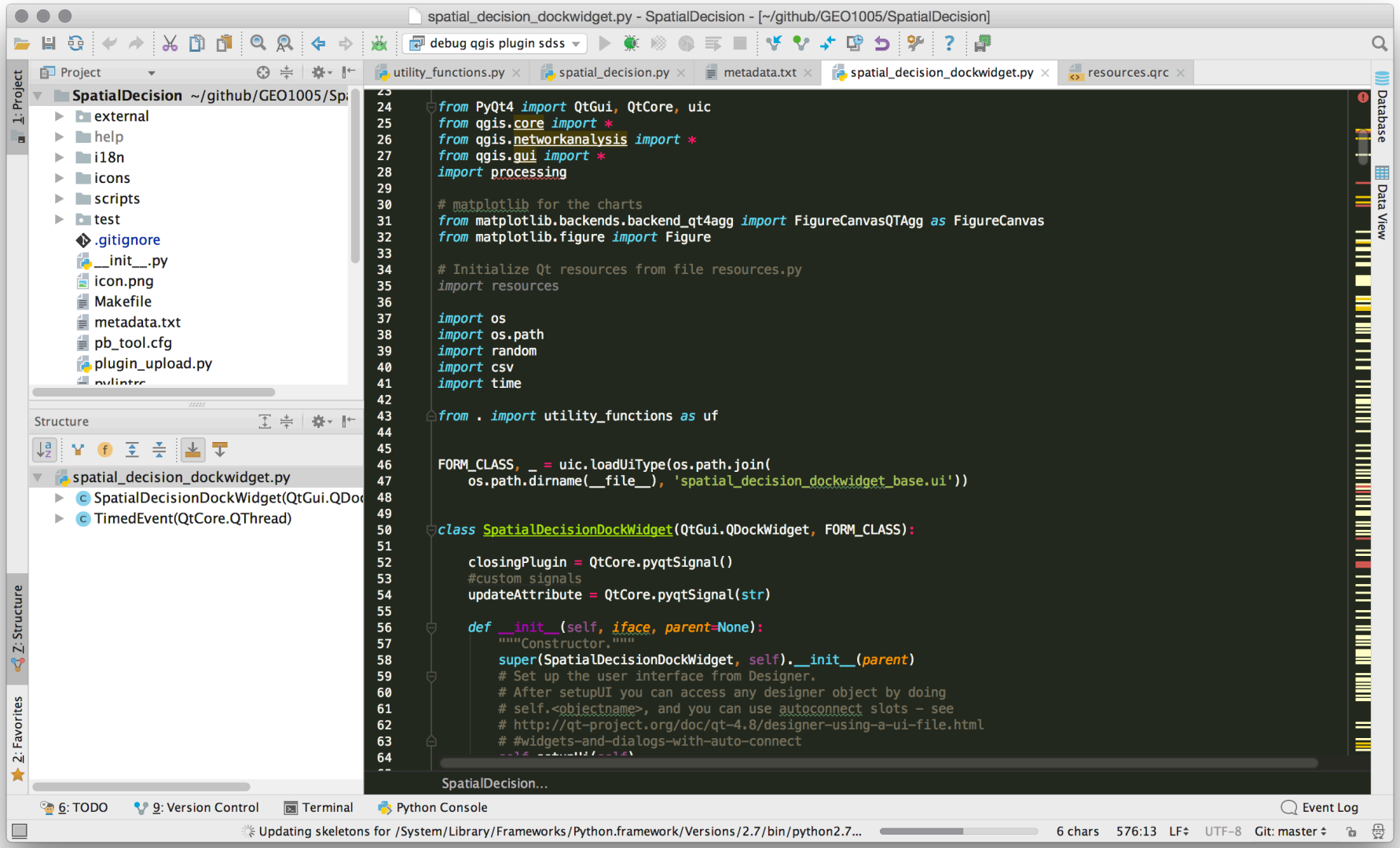
The free Community version does not have the remote debugger. (more on this later)

The Pro version has a 30 days trial, but students can obtain a free license by registering using their academic TU Delft email.

Download and install the version for your system:

https://www.jetbrains.com/pycharm/download/

Eclipse with PyDev is an alternative Python IDE, but only use it if you're very familiar with it…


spatial_decision_dockwidget.py - SpatialDecision - [~/github/GEO1005/SpatialDecision]
debug qgis plugin sdss
utility_functions.py
spatial_decision.py
metadata.txt
spatial_decision_dockwidget.py
resources.qrc
Project
SpatialDecision ~/github/GEO1005/Spa
external
help
i18n
icons
scripts
test
.gitignore
__init__.py
icon.png
Makefile
metadata.txt
pb_tool.cfg
plugin_upload.py
Structure
spatial_decision_dockwidget.py
SpatialDecisionDockWidget(QtGui.QDoc
TimedEvent(QtCore.QThread)
from PyQt4 import QtGui, QtCore, uic
from qgis.core import *
from qgis.networkanalysis import *
from qgis.gui import *
import processing
# matplotlib for the charts
from matplotlib.backends.backend_qt4agg import FigureCanvasQTAgg as FigureCanvas
from matplotlib.figure import Figure
# Initialize Qt resources from file resources.py
import resources
import os
import os.path
import random
import csv
import time
from . import utility_functions as uf
FORM_CLASS, _ = uic.loadUiType(os.path.join(
os.path.dirname(__file__), 'spatial_decision_dockwidget_base.ui'))
class SpatialDecisionDockWidget(QtGui.QDockWidget, FORM_CLASS):
closingPlugin = QtCore.pyqtSignal()
#custom signals
updateAttribute = QtCore.pyqtSignal(str)
def __init__(self, iface, parent=None):
"""Constructor."""
super(SpatialDecisionDockWidget, self).__init__(parent)
# Set up the user interface from Designer.
# After setupUI you can access any designer object by doing
# self.<objectname>, and you can use autoconnect slots - see
# http://qt-project.org/doc/qt-4.8/designer-using-a-ui-file.html
# #widgets-and-dialogs-with-auto-connect
SpatialDecision...
Database
Data View
1: Project
7: Structure
2: Favorites
6: TODO
9: Version Control
Terminal
Python Console
Event Log
Updating skeletons for /System/Library/Frameworks/Python.framework/Versions/2.7/bin/python2.7...
6 chars
576:13
LF
UTF-8
Git: master

(Windows only) Configure PyCharm start command

In Windows you need to tell PyCharm the location of the Python interpreter that comes with QGIS.

- Download this batch file from Brightspace and edit:

- Change the OSGEO4W_ROOT variable path to your OSGeo4W installation path
- Change the QGIS and PyCharm application paths
- Save the file
- **ALWAYS** start PyCharm using this shortcut

```
@echo off
SET OSGEO4W_ROOT=C:\OSGeo4W64

call "%OSGEO4W_ROOT%"\bin\o4w_env.bat

path %PATH%;%OSGEO4W_ROOT%\apps\qgis-ltr\bin

set PYTHONPATH=%PYTHONPATH%;%OSGEO4W_ROOT%\apps\qgis-ltr\python;
set PYTHONPATH=%PYTHONPATH%;%OSGEO4W_ROOT%\apps\Python27\Lib\site-packages
set QGIS_PREFIX_PATH=%OSGEO4W_ROOT%\apps\qgis-ltr

start "PyCharm aware of Quantum GIS" /B "C:\Program Files (x86)\JetBrains\PyCharm 2016.2.3\bin\pycharm.exe" %*
```

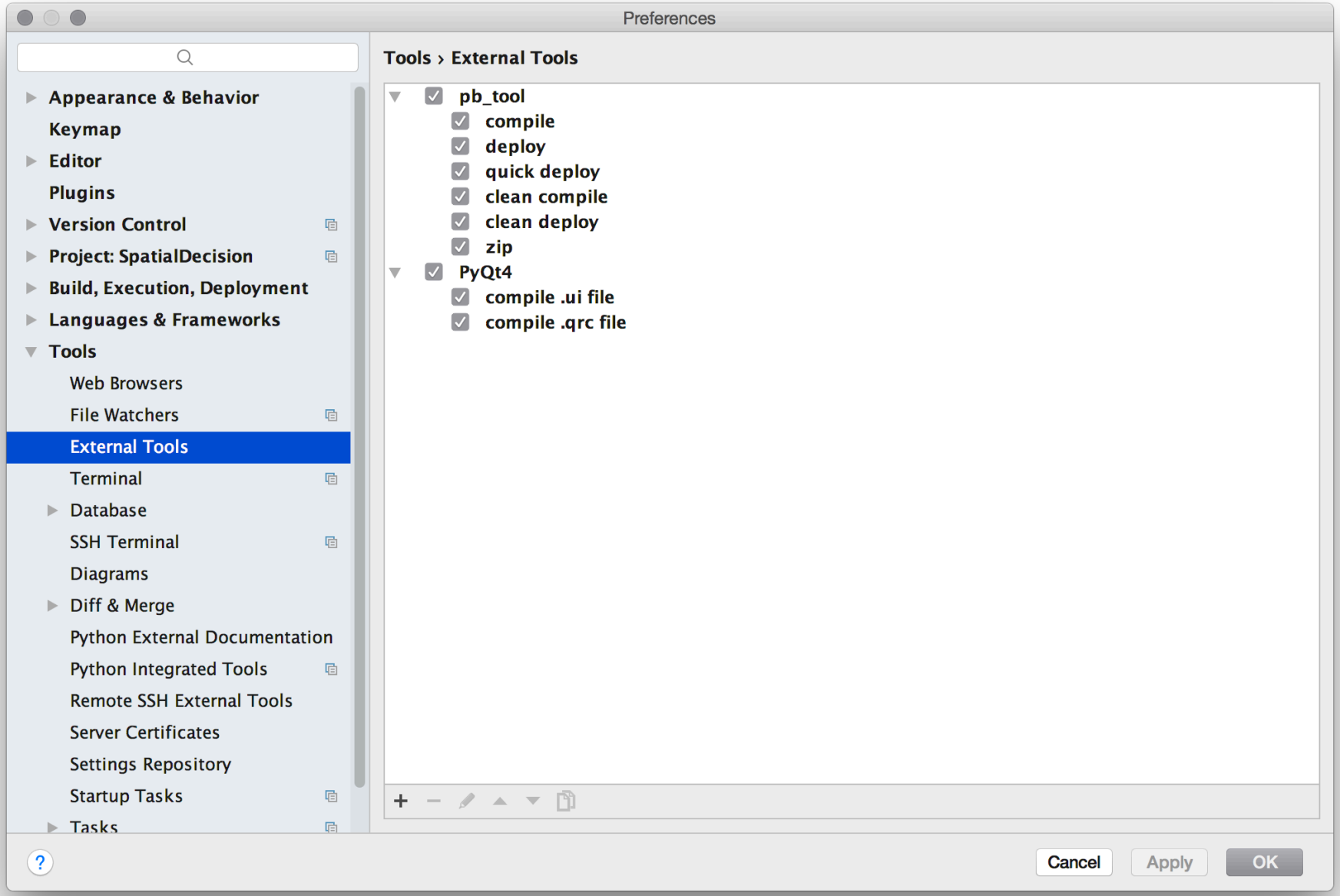
To configure PyCharm to use the pb_tools:

- Click “Configure > Preferences…” in the Welcome window or
- Go to “File > Settings...” (Windows) or
- Go to “PyCharm > Preferences…” (Mac OSX)

Configuration tasks:

- Create ‘External Tools’ for compiling and deployment
- Add ‘External Tools’ tools to menus and toolbar
- Create a QGIS aware PyCharm command
- Create a QGIS remote debug configuration

Create ‘External Tools’ for pb_tool

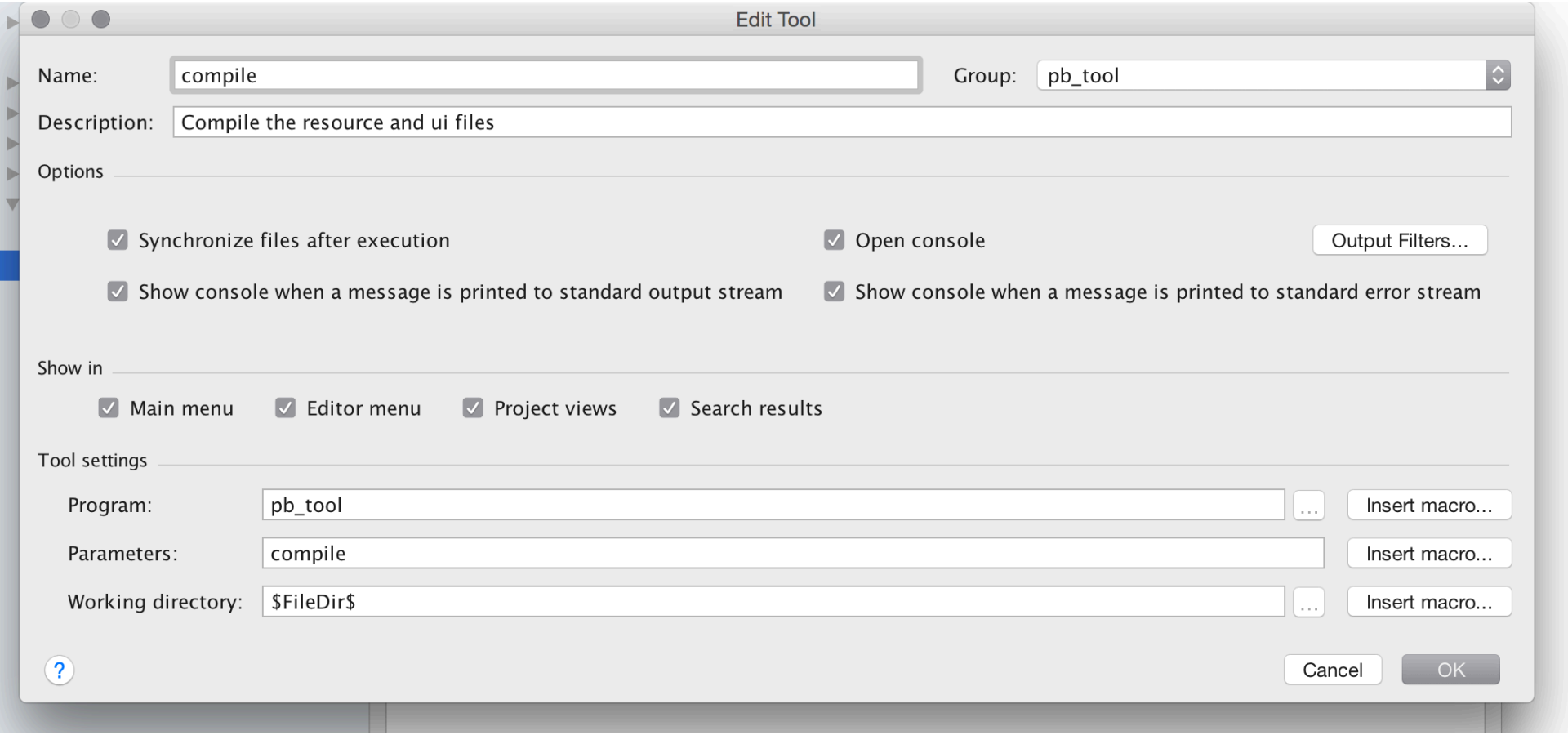


pb_tool has a collection of commands that make it very easy to make plugins available in QGIS for use and testing.

- **Compile** – process resources and ui files that need to be converted to python.
- **Deploy** – compile and copy the relevant plugin files into the QGIS plugins directory.
- **Quick deploy** – only updates the plugin files, it’s faster than deploy
- **Clean compile** – deletes the compiled files.
- **Clean deploy** – deletes the plugin from the QGIS plugins directory.
- **Zip** – creates a zip file with the relevant plugin files for distribution.

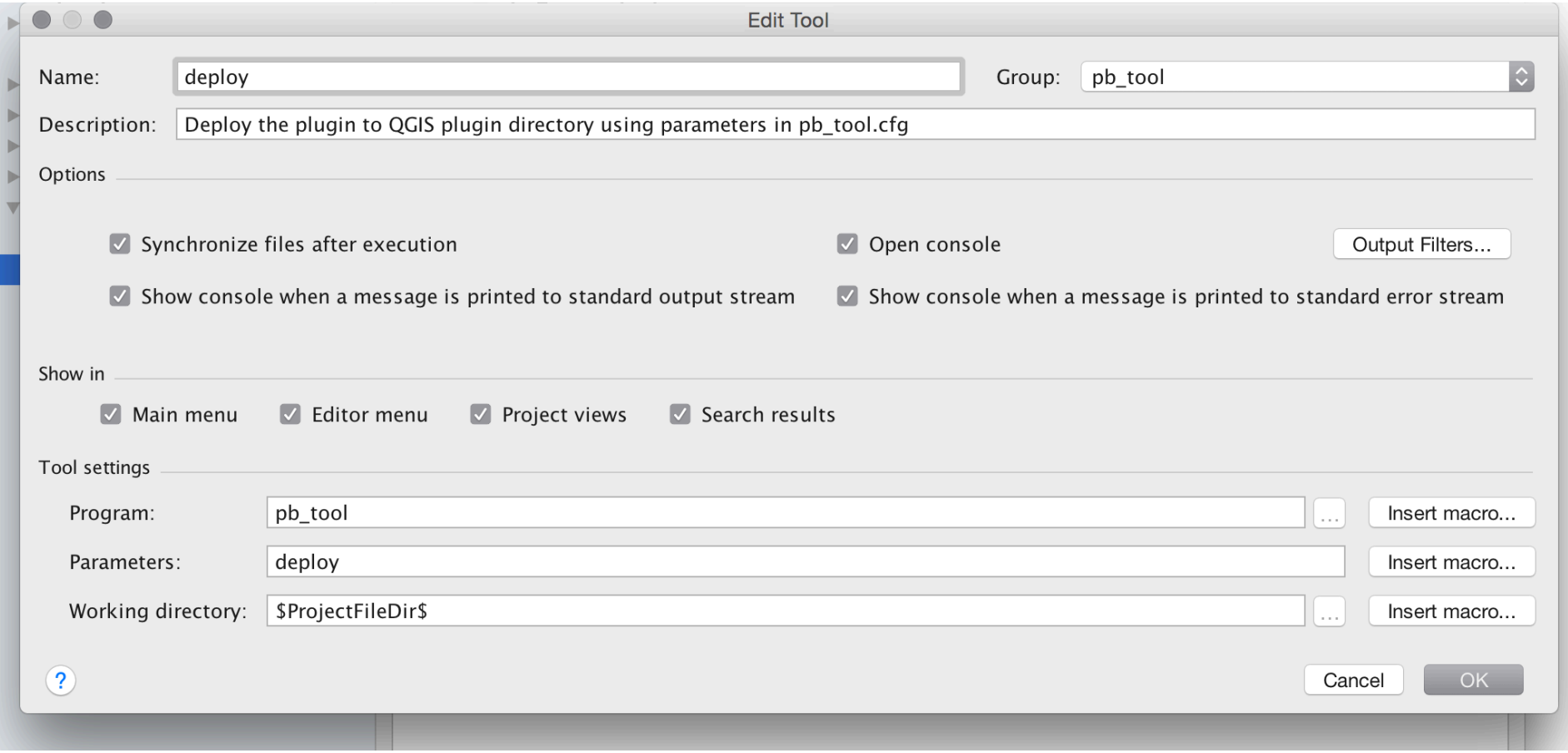
Set-up pb_tool - compile

- Compiles the user interface and resources files.
- Only required ocasionaly when those change



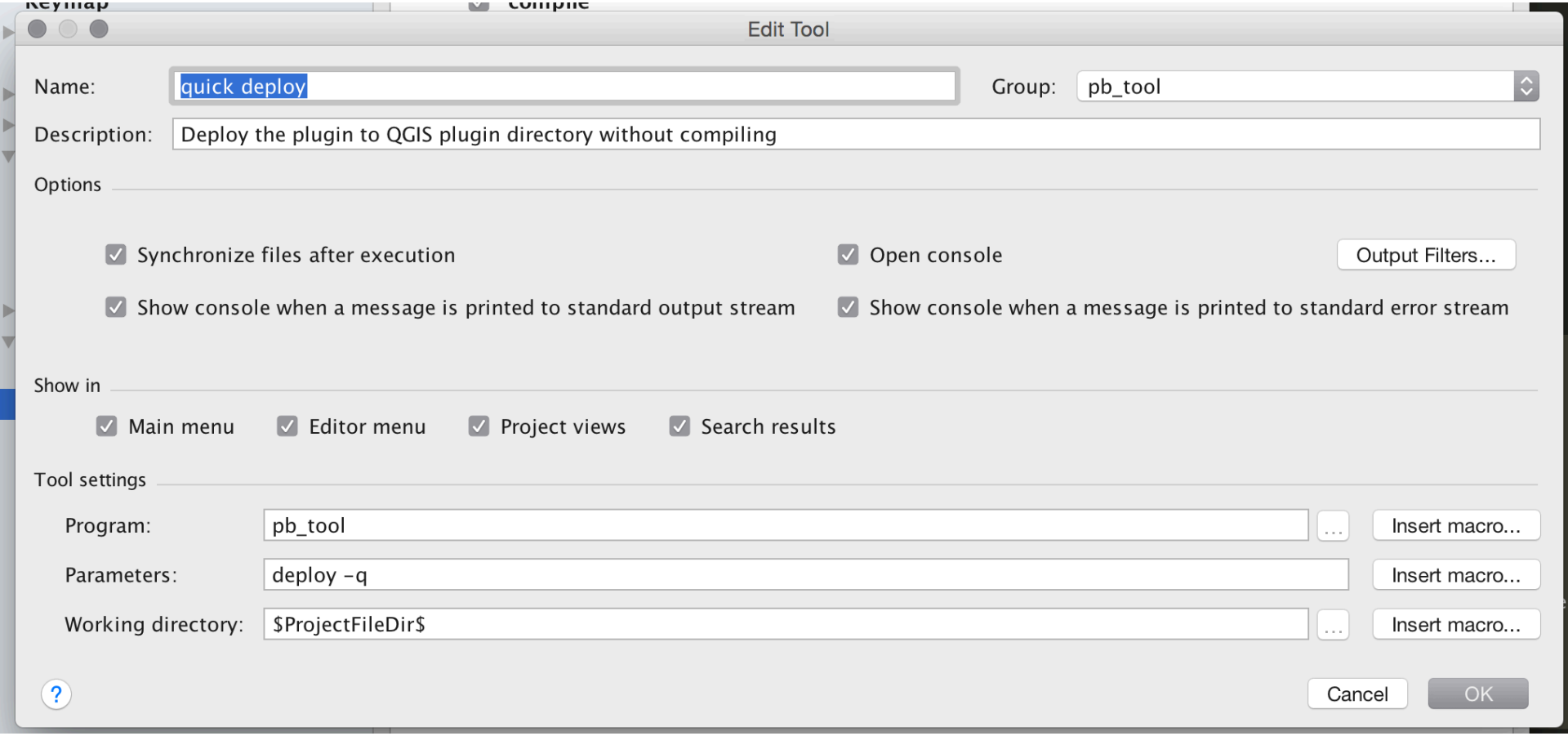
Set-up pb_tool - deploy

- Prepares the project and copies the plugin files into the plugins folder.
- Slower and asks the user for confirmation.



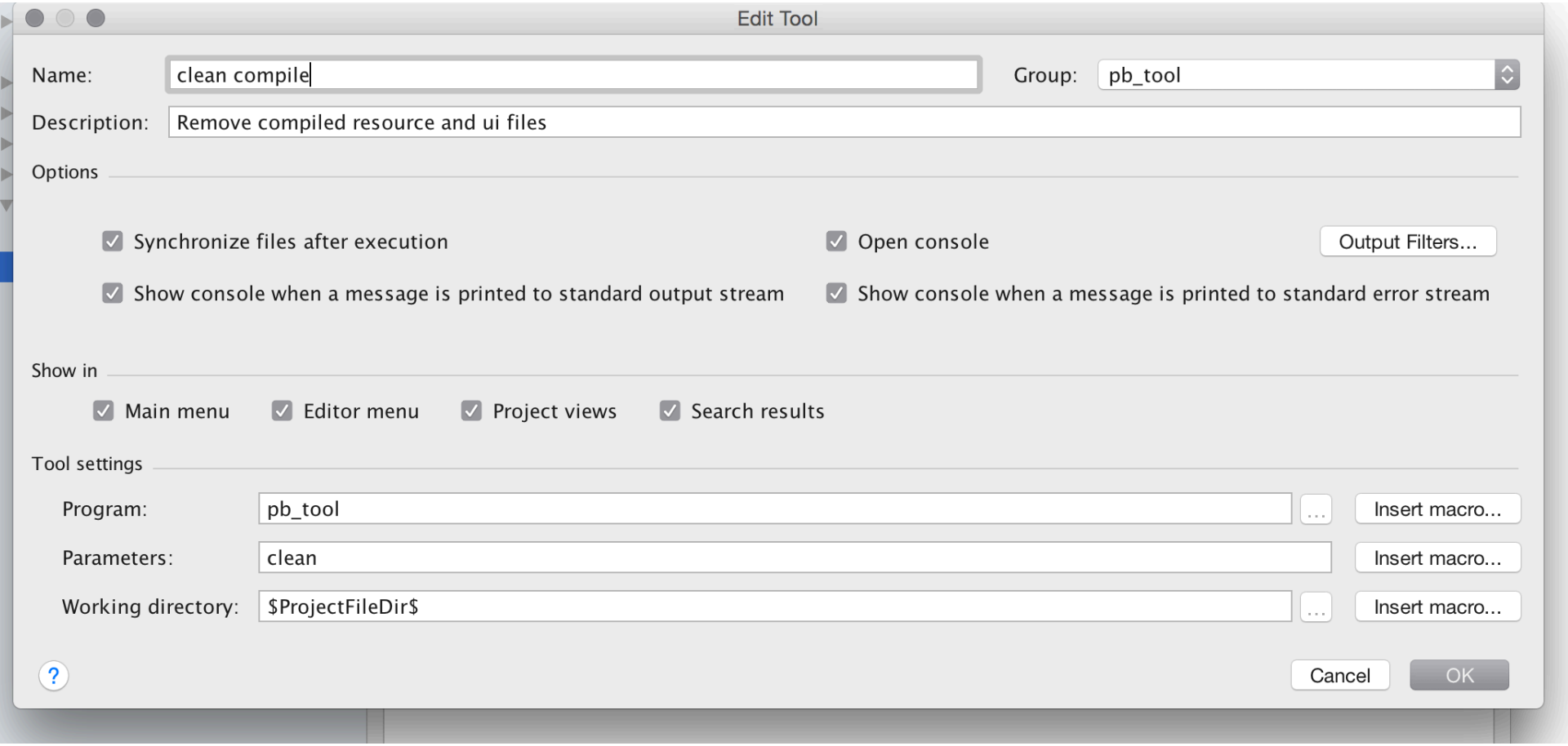
Set-up pb_tool – quick deploy

- Updates the project in the plugins folder, only copying relevant files.
- Does not require user intervention.



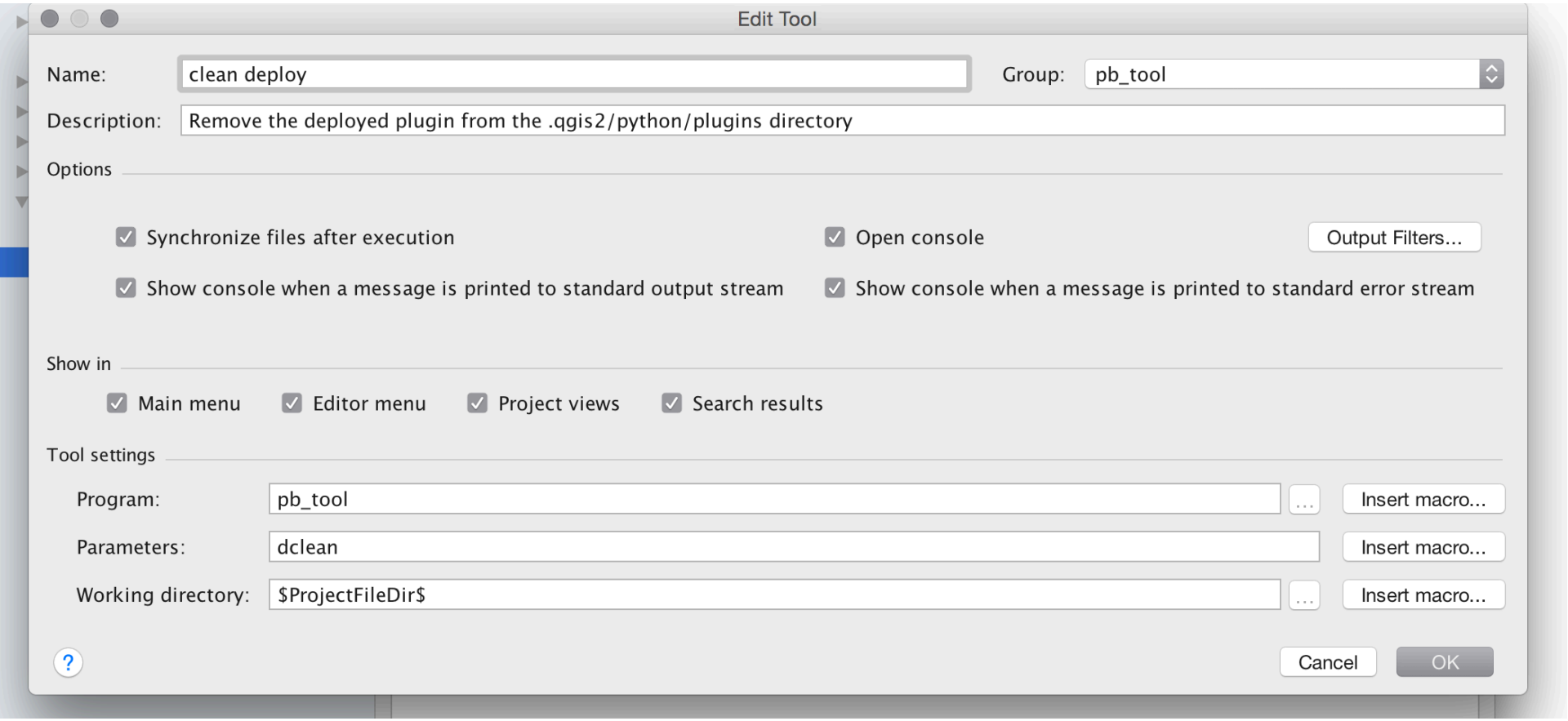
Set-up pb_tool – clean compile

- Removes the previously compiled resources and compiled ui files

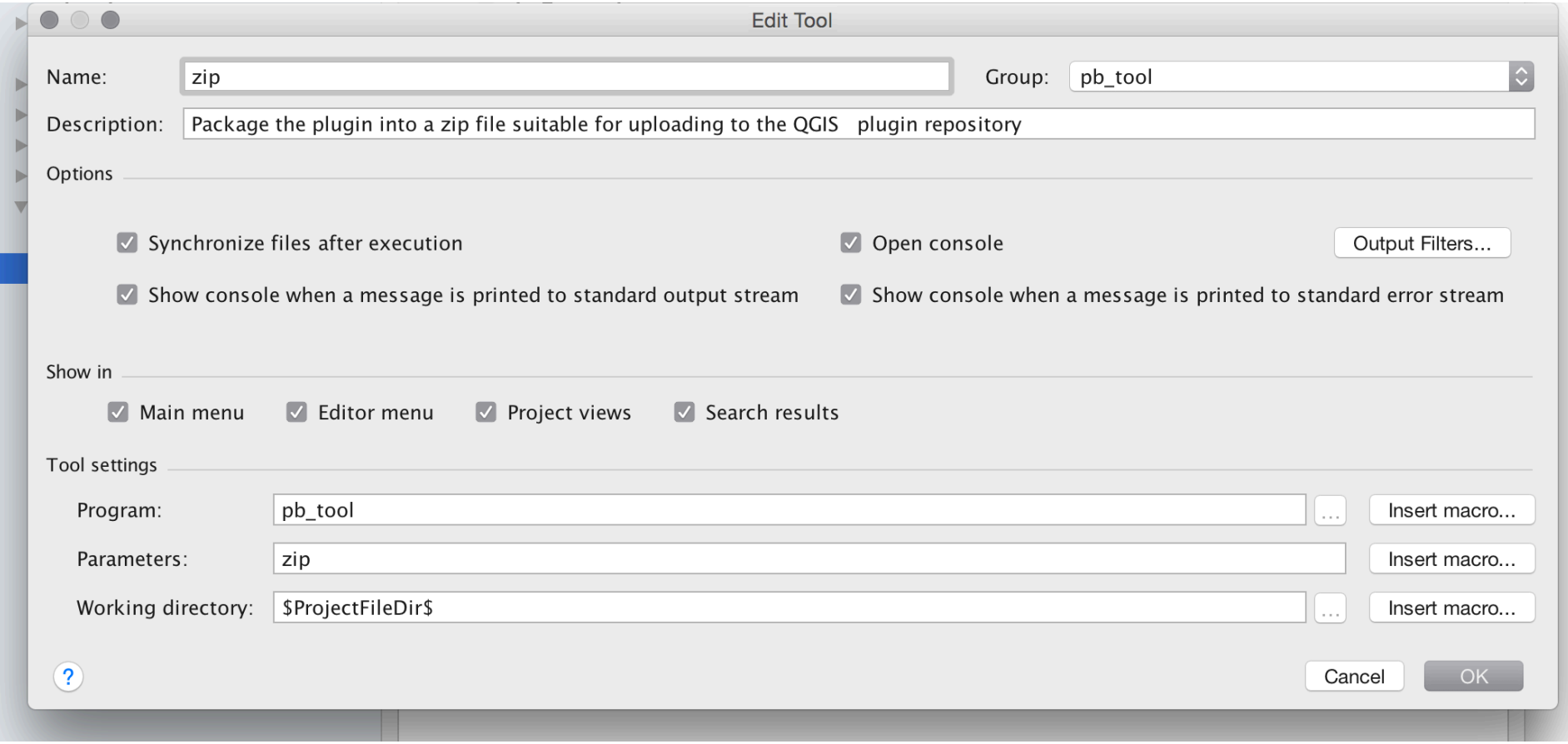


Set-up pb_tool – clean deploy

- Removes the previously deployed plugin from the QGIS plugins folder

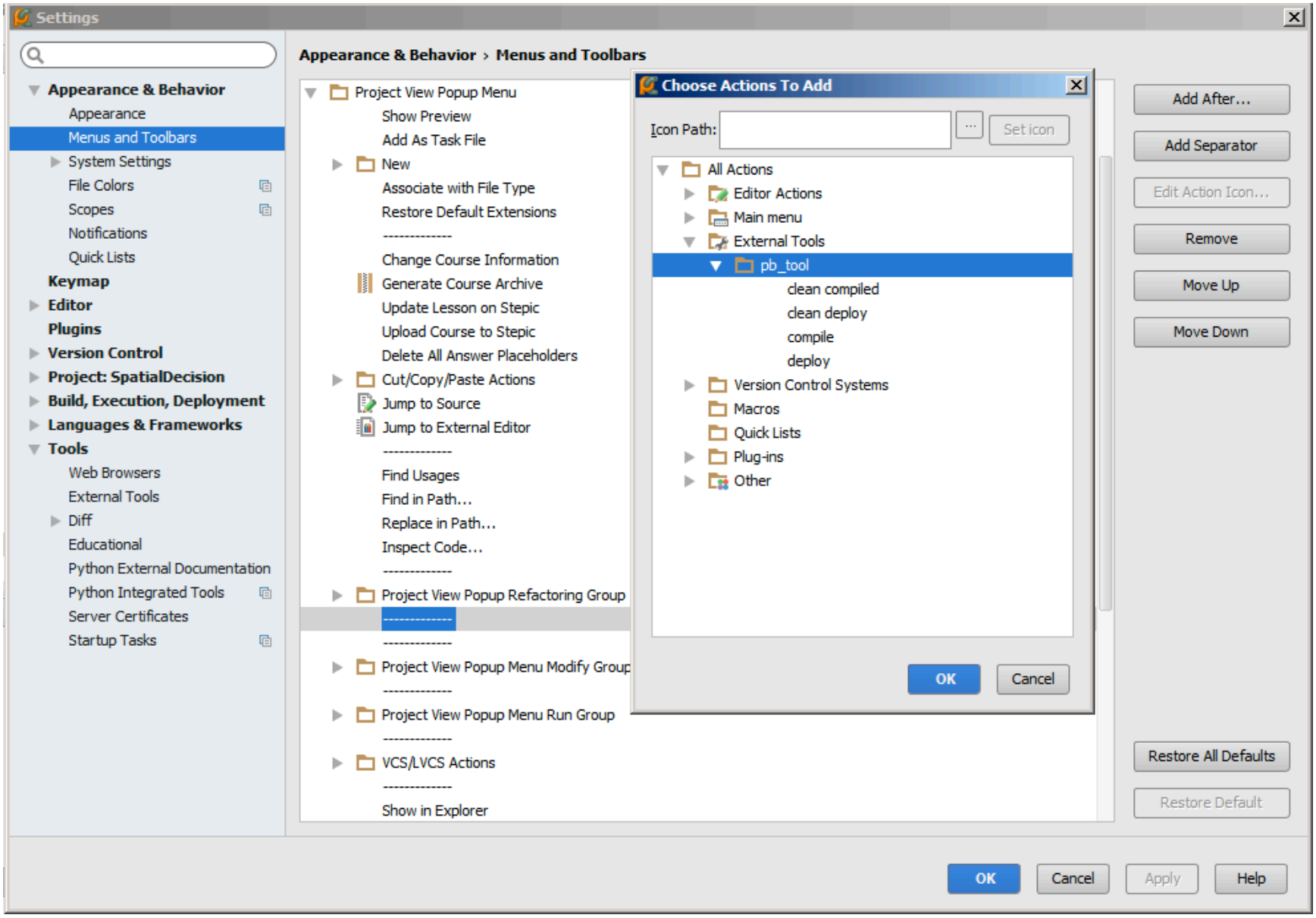


- Creates a clean zip file of the plugin for distribution



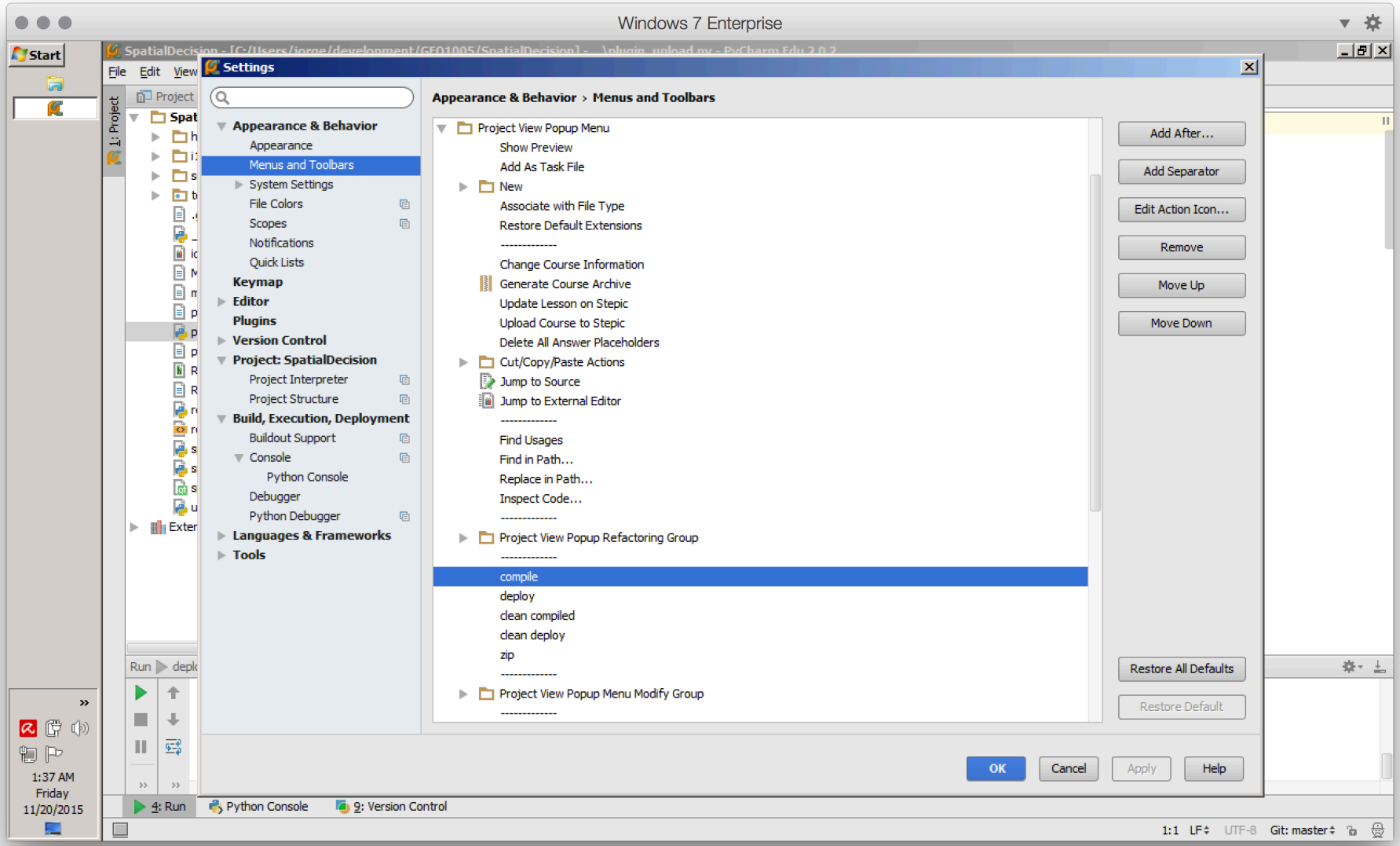
- Click ‘Apply’ after creating the External Tools

Add ‘External Tools’ to menus



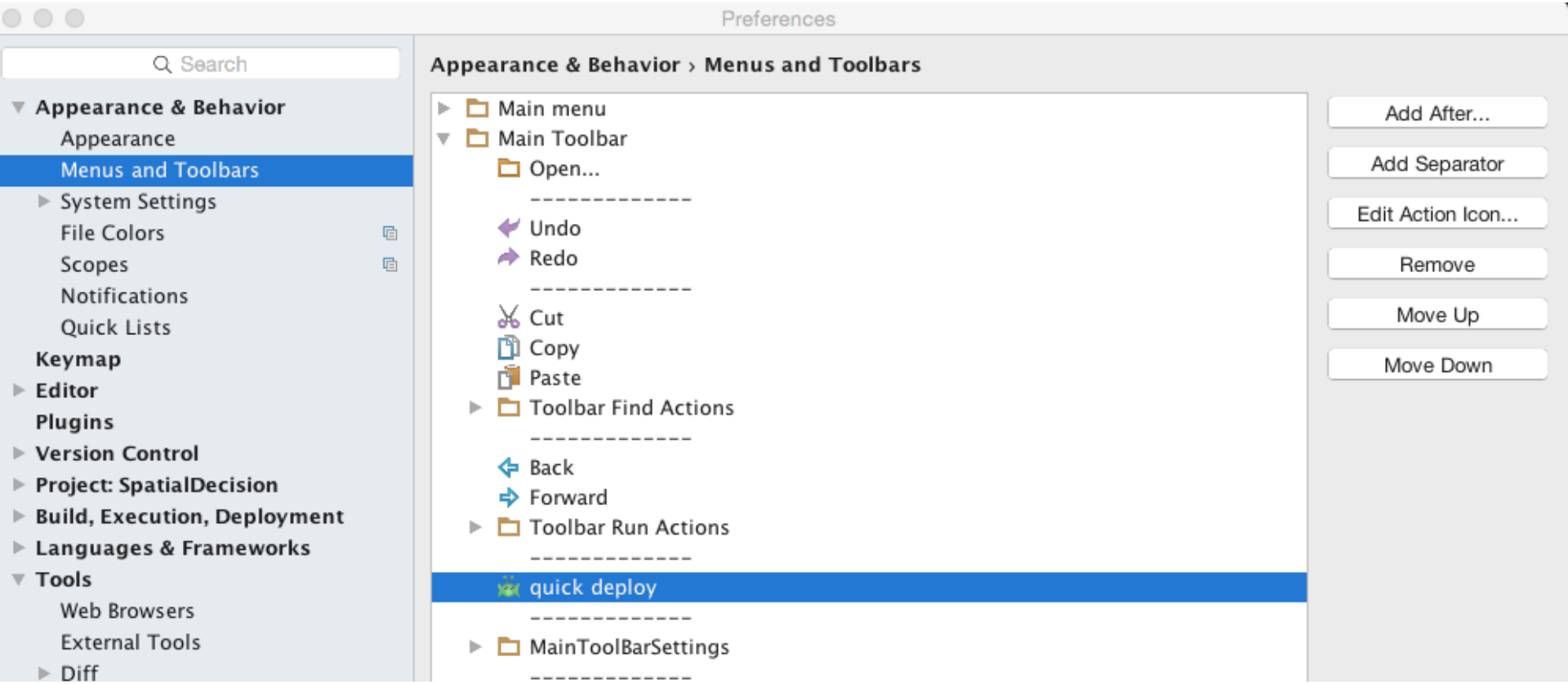
Add ‘External Tools’ to menus

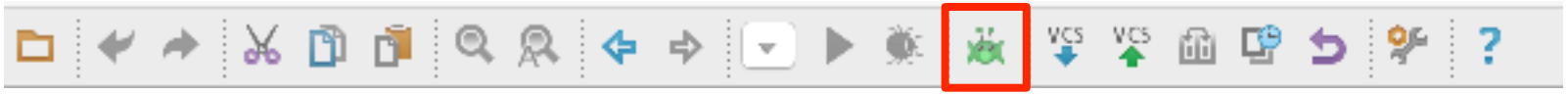
- Add individual tools or the ‘pb_tool’ group



Add ‘Quick deploy’ to the main toolbar

- Add button next to the debug button, for example.
- Click ‘Ok’ when finished.





Install the pydevd package (Pycharm Pro only)

Install the pydevd Python package to allow remote debugging in PyCharm Pro.

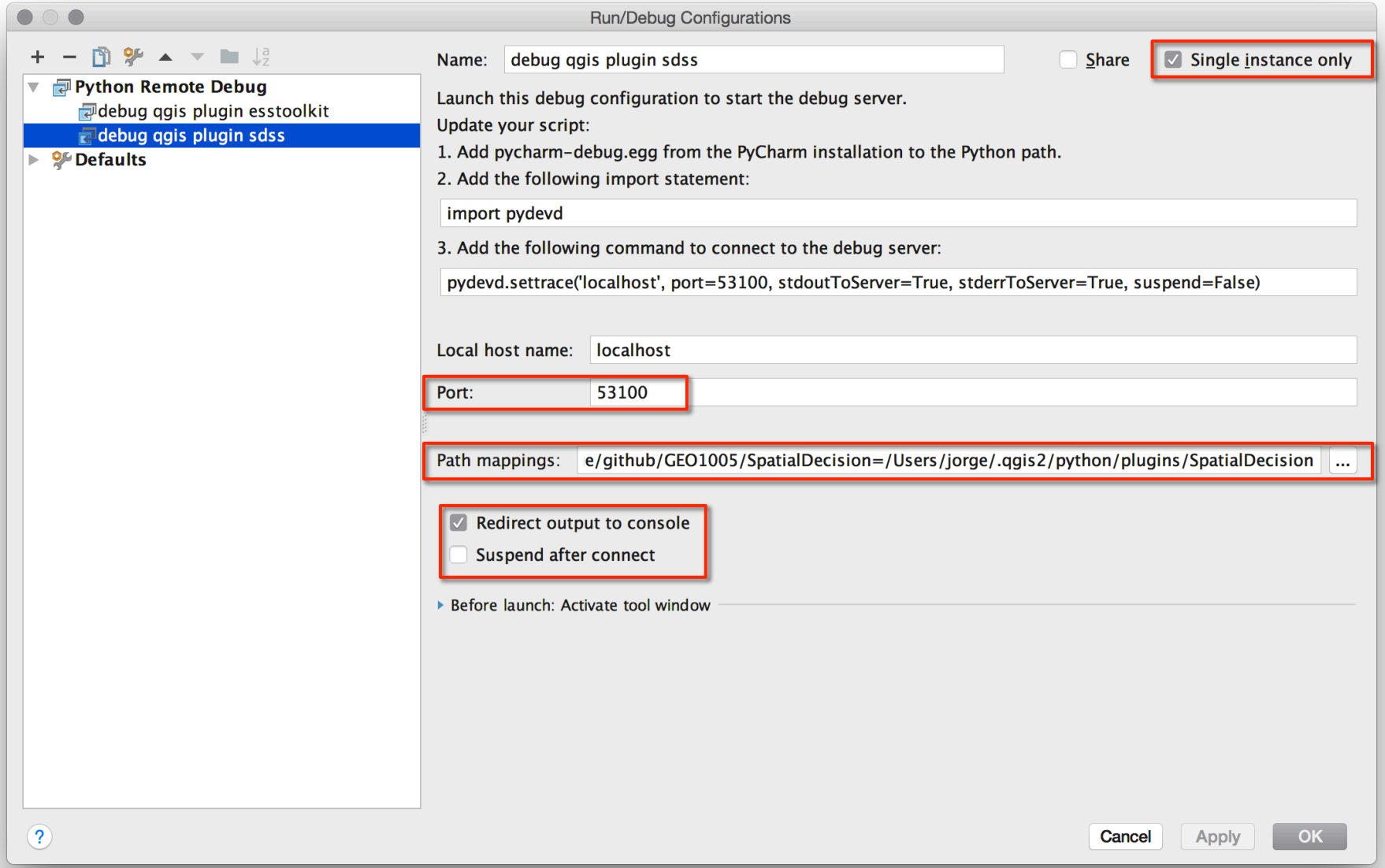
- Locate the egg file in the Pycharm installation folder:
  - Windows: \Program Files (x86)\JetBrains\PyCharm xxxxx\debug-eggs\pycharm-debug.egg
  - OS X: Applications/PyCharm/Contents/debug-eggs/pycharm-debug.egg
- Copy the pycharm-debug.egg file to some location
- Start the OSGeo4W Shell or the OS X Terminal
- Change to the 'pycharm-debug.egg' location
  - **`Cd`** …
- Install the debugger using:
  - **`python -m easy_install pycharm-debug.egg`**
- Or if you're on Mac OSX:
  - **`sudo python -m easy_install pycharm-debug.egg`**

Create ‘Remote Debug’ configuration (Pycharm Pro only)

In PyCharm go to menu ‘Run’ > ‘Edit Configurations…’
(See screenshot on next slide)

- Click the ‘+’ sign to add a configuration
- Select ‘Python Remote Debug’
- Give a name to the configuration, e.g. project name debug
- Check ‘Single instance only’
- Change the port to ‘53100’
- Uncheck ‘Suspend after connect’
- Add path mappings, between the source code folder and the deployed plugin folder (in .qgis2)
  - Click ‘…’ and type the path to the source (left) and plugin (right)
- Click ‘Ok’

Create ‘Remote Debug’ configuration (Pycharm Pro only)



Ready to develop QGIS plug-ins!

(at last…)