An Experience in Using Python

with Anaconda, Eclipse, Python2, Python3, GUI Programming

Ray Smith ([raysmith@alum.mit.edu](mailto:raysmith@alum.mit.edu))

April 25, 2018

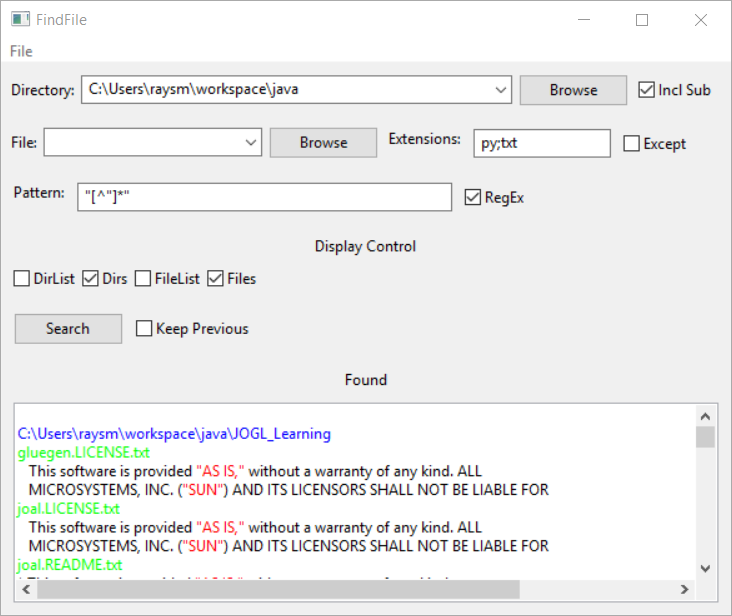
# Introduction

Python, while not the only programming language in use today, is a rather popular and powerful tool. My goal today is to present my latest experience, using Python combined with some other useful tools, giving the audience some insight in to their use in program development. All of these tools are worthy of talks, books, or full courses on themselves. My plan is to touch on each, exposing some benefits, including some pointers affording individuals the opportunity of proceeding further investigation as they may see fit.

Ray Smith (Charles Raymond Smith)

* Decades of Programming – engineering, scientific, financial
* Programming environments – embedded, systems, application
* Languages – C, C++, Perl, Java, Python, Assembly, shell, Fortran, PL1, …
* Currently investigating graphics / game programming

An Example – fileFind – a file text searching tool

* Windows GUI version of Unix/Linux’s **find**
* Reduced function but too simple
* A work in progress
* <https://github.com/raysmith619/findFile/>
* 

findFile.py Screen Shot

* Features – so far
  + Directory/File Searching
  + Subdirectory search
  + Multiple file extensions
  + Regular expression text patterns
  + Colorized text accents
  + Directory / File setting history
  + Settings restored between program runs
* Features contemplated
  + Easier
    - Multiple patterns
    - Listing multiple occurrences within a line
    - Multiline patterns
    - Output formatting with match patterns
  + Less so
    - Additional file types e.g. MsWord, MsExcell
    - Searching search results

Python – Programming language

* popular programming language in wide use
* Used in Data Analytics
* Used as introduction to programming at MIT

Python2, Python3

* Two dialects of the Python Language
* Independent lines – 2.x, 3.x
* Python2.x – lots of legacy code, faster?...
* Python3.x – possibly the future, language improvements
* Python2.x – print “Hi..”, except Exception, e:
* Python3.x – print(“Hi…), except Exception as e:
* <https://www.digitalocean.com/community/tutorials/how-to-port-python-2-code-to-python-3>
* <http://python-future.org/overview.html>

Anaconda – Installation / Development System

* <https://www.anaconda.com/what-is-anaconda/>
* “With over 6 million users, the open source [Anaconda Distribution](https://www.anaconda.com/distribution/) is the easiest way to do Python data science and machine learning. It includes 250+ popular data science packages and the ***conda*** package and virtual environment manager for Windows, Linux, and MacOS. …”
* High-Performance Distribution
* Package Management
* Python 3.6 version
* Python 2.7 version

https://docs.anaconda.com/anaconda/install/windows

Eclipse – Graphical User Interface (GUI)

* Development Platform for Java, Python, C/C++, Perl
* Powerful Debugger
* Project Maintenance

GUI Programming

* The world is *GRAPHICAL*
* Too much to do from scratch
* Similarities across OS / Language
* We are trying out wxPython
* We have used others, e.g. **Tkinter**

Properties Files - A simple Python 2 to 3 experience

* Properties files – heavily used in Java
* Provide setting / controls memory between executions
* Simple formatting: “name = value”

Properties Module Transition

1. p\_1.py – original off the WEB
2. p\_2.py – after a few Python2.x to Python3.x changes
3. p\_3.py – after a few Python2.x to Python3.x changes
4. p\_4.py – Coding to make a better test
   1. A more sophisticated P2.x 🡺 P3.x issue
5. p\_5.py – Completing a few more changes

findFile - A GUI Example - Learning some wxPython

* **wxPython tutorial -** [**http://zetcode.com/wxpython/**](http://zetcode.com/wxpython/)

“wxPython is a cross platform toolkit for creating desktop GUI applications. The principal author of wxPython is Robin Dunn. With wxPython developers can create applications on Windows, Mac and on various Unix systems. wxPython is a wrapper around wxWidgets, which is a mature cross platform C++ library.”

* Examples were cut and pasted into Python files
* Absent from the tutorial are “.png” graphics files

Graphic files were created manually

findFile.py Structure

Files:

* findFile.py – main file
* filebrowsebutton.py
  + From the NET
  + implements directory / file specification / input
* properties.py
  + From the NET
  + read, write, parse properties files

findFile.py functions

**InitUI**

* Setup GUI panel view
* Setup search input fields
* Setup input update actions
* Setup found display

**search**

* Check / Record Input settings
* Collect selected Directories / Files
* Traverse files
* Format and Display Results

**on…**

* Perform action based on user input action

**on…Change**

* Update input field based on user input action

**getProperty**

* Retrieve setting based on Properties file
* Setting may be updated

**setProperty**

* Store value for saving in Properties file
* Values updated, but saved at a later time

**propSave**

1. Save properties to file

Properties Processing

For each value needing persistence - property

1. Define property key string e.g. PK\_DIR\_START
   * Often of the form name1.name2.name3
2. Check if present, if not give default value
3. Split value if segmented e.g. *value1*;*value*2;…
4. Set input field
5. Simulate change, calling **on…Change**
6. ..
7. Field changes invoke **on…Change** call
   * Update internal value
   * Save property value

For properties as a group

* Load properties from file
* At end, save properties to file

Data Field Processing

For each data item

1.Design Data Field

* Including type, position, …
* Including nesting in field
* Initialize field

2.Bind changing event to **on…Change** function

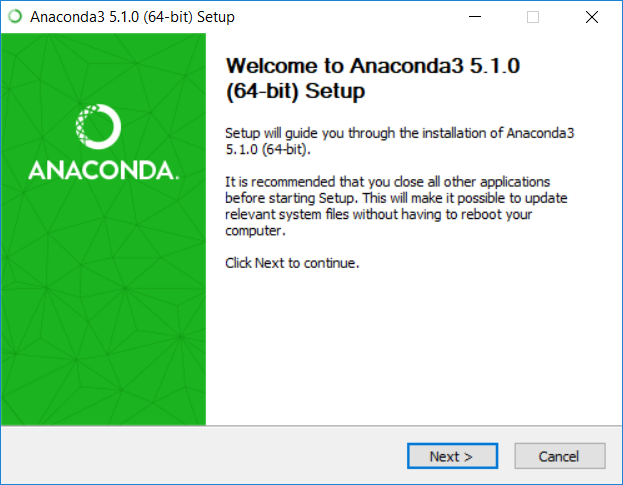
3.Define on…Change function to

* Update internal value(s)
* Update property value
* Do any checking and updating other values

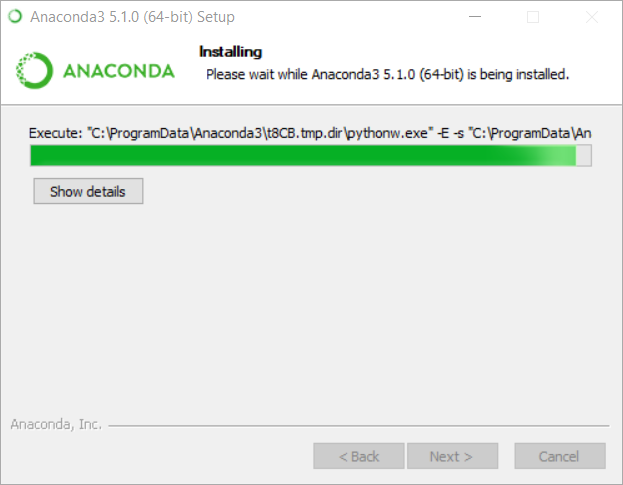
Installing Anaconda

<https://conda.io/docs/user-guide/install/download.html>

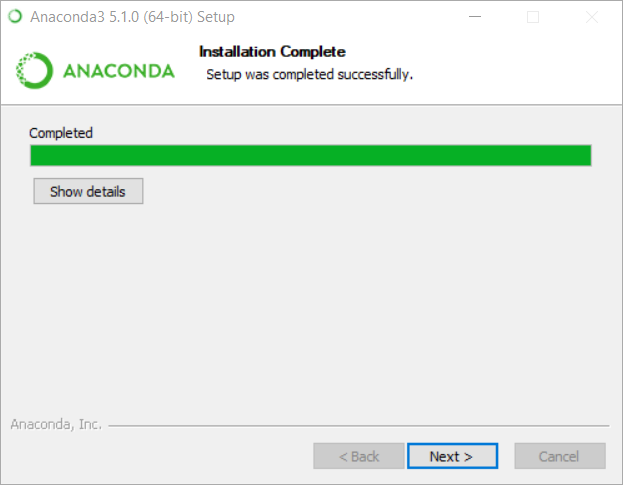
* NOT BOTH Anaconda2 and Anaconda3
* Install one, switch between Python2/3 if necessary
* Example: Anaconda3-5.1.0-Windows-x86\_64.exe



Startup window, executing above …\_64.exe

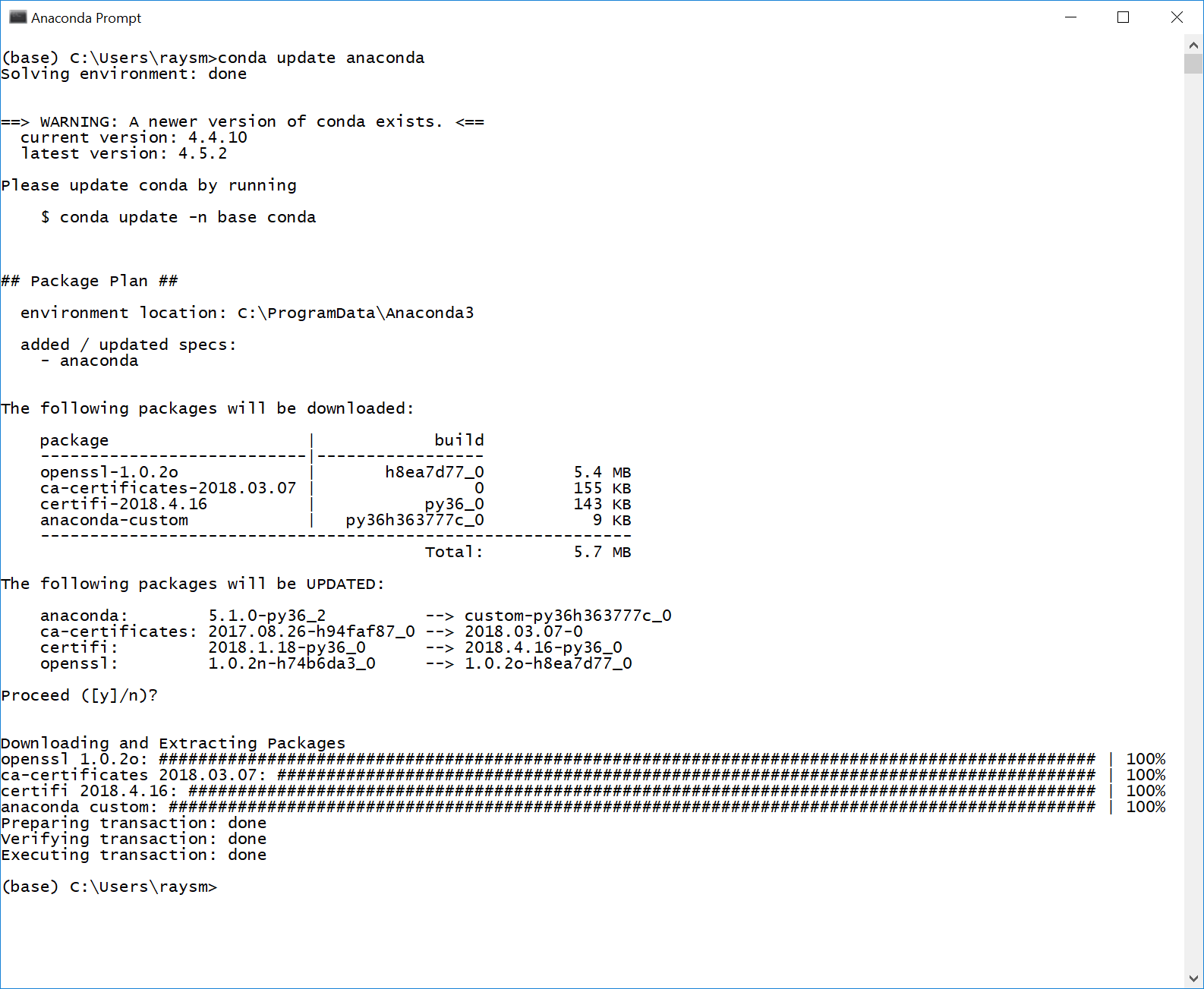


Processing – Takes minutes

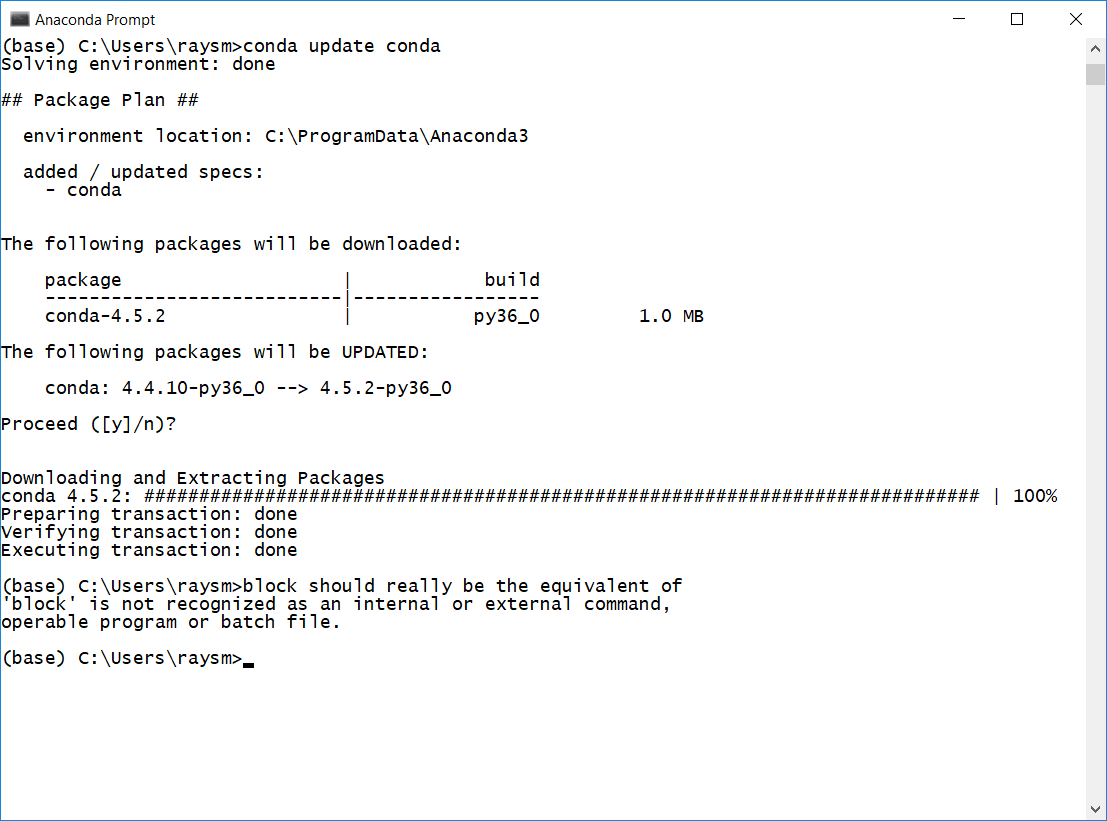


At end of installing

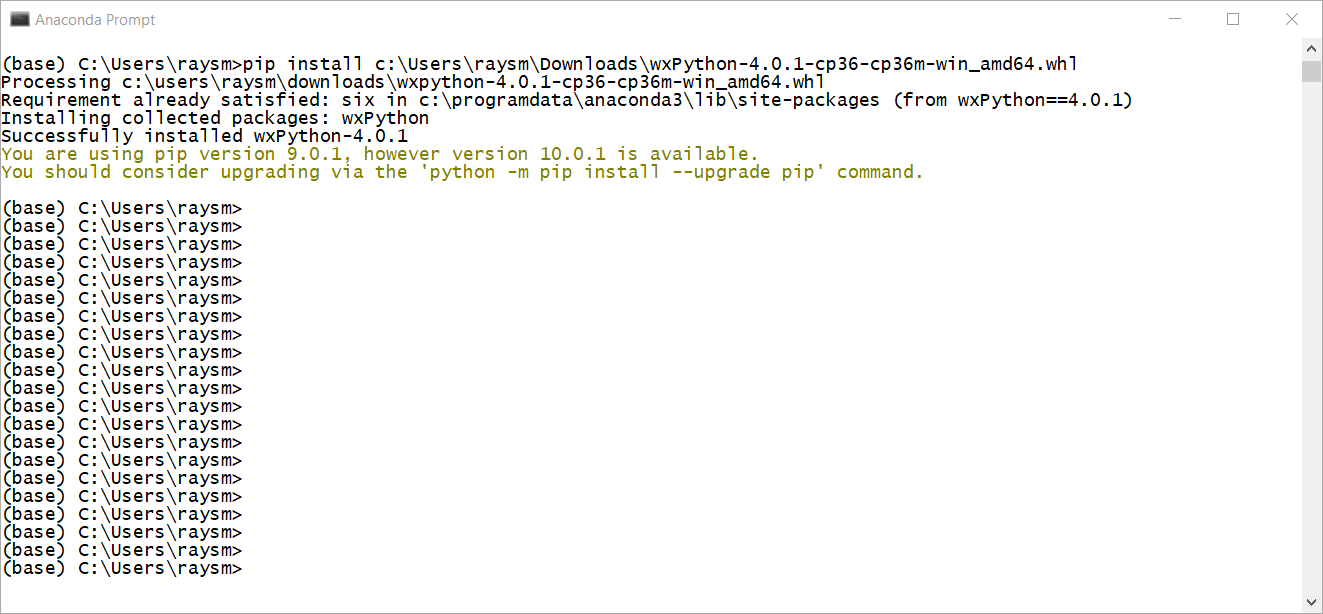
Updating Anaconda



Update conda command



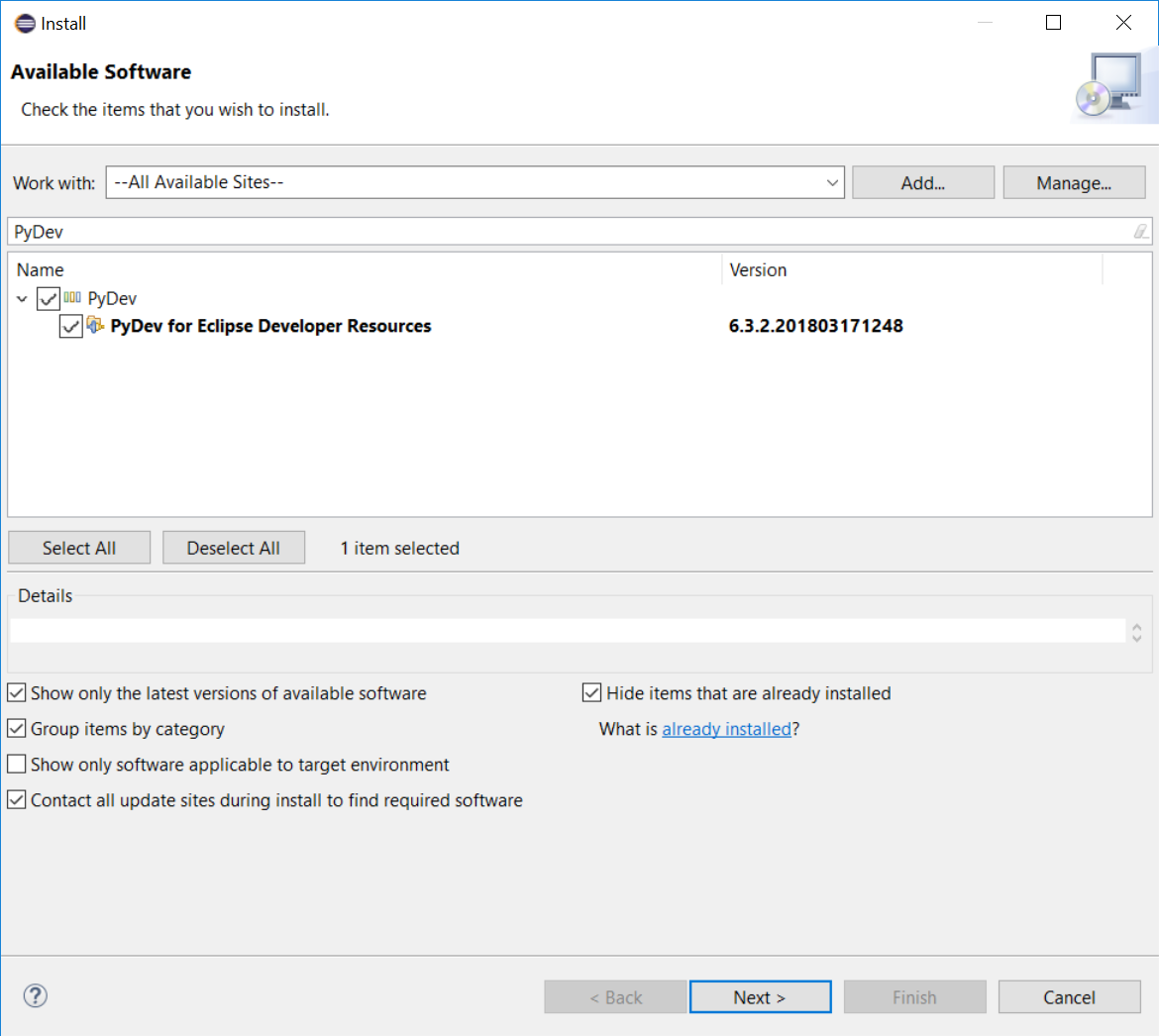
Adding Graphics Support (wxPython) via Anaconda Shell



Anaconda Prompt view

Adding / Updating Eclipse with Python Development (PyDev)

Eclipse Menu: Help 🡪 Install New Software



Eclipse Available Software after setting sites, Filter Name

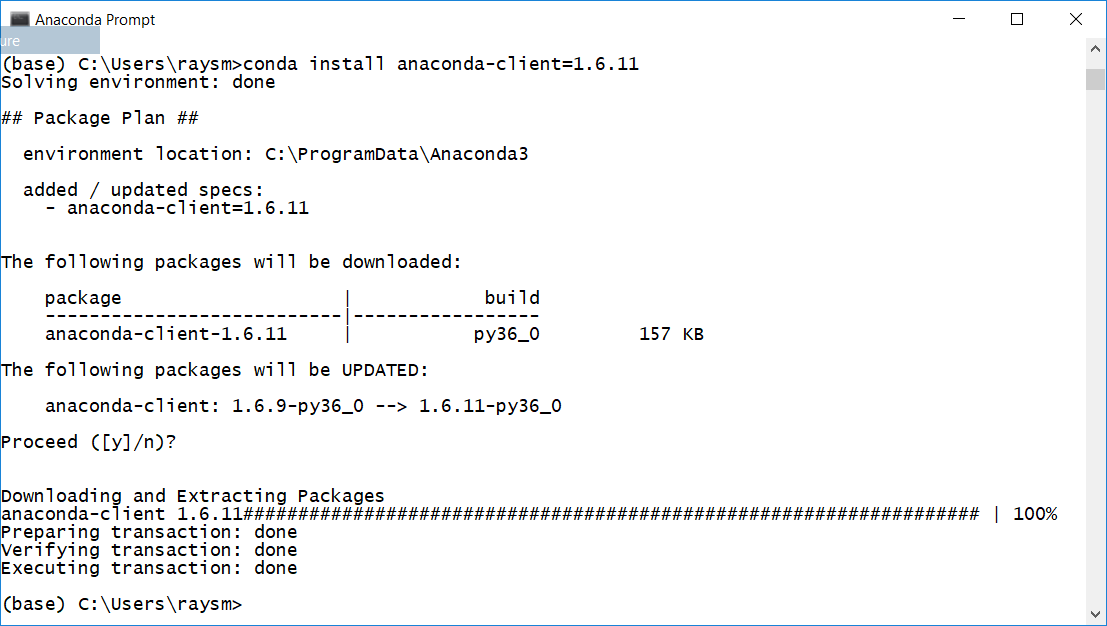
Click Next, Agree with License, …

Creating Anaconda environments

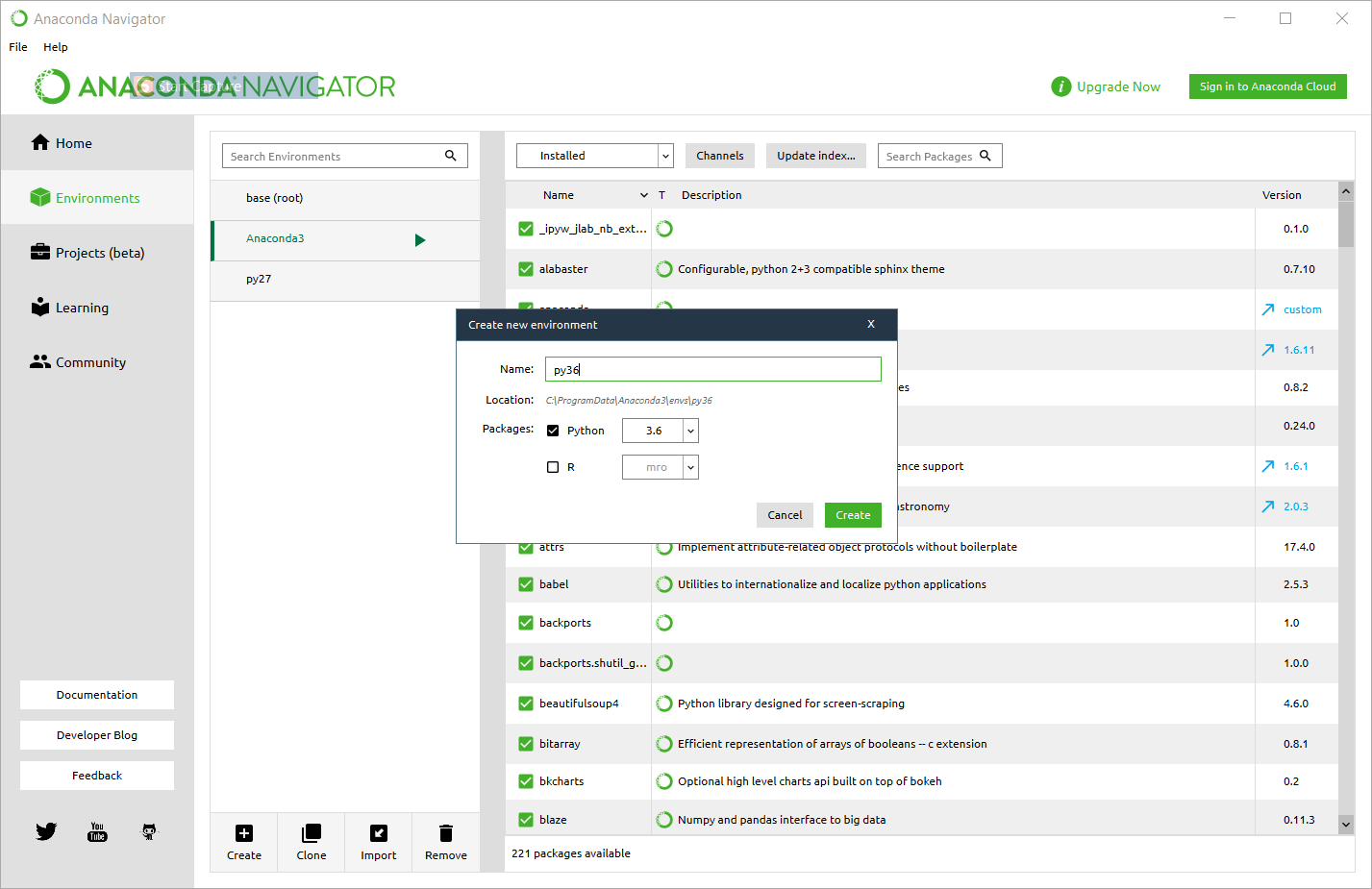
Using Anaconda Navigator (from Programs)

**Got error**: could not determine a constructor for the tag 'tag:yaml.org,2002:python/unicode' #8568

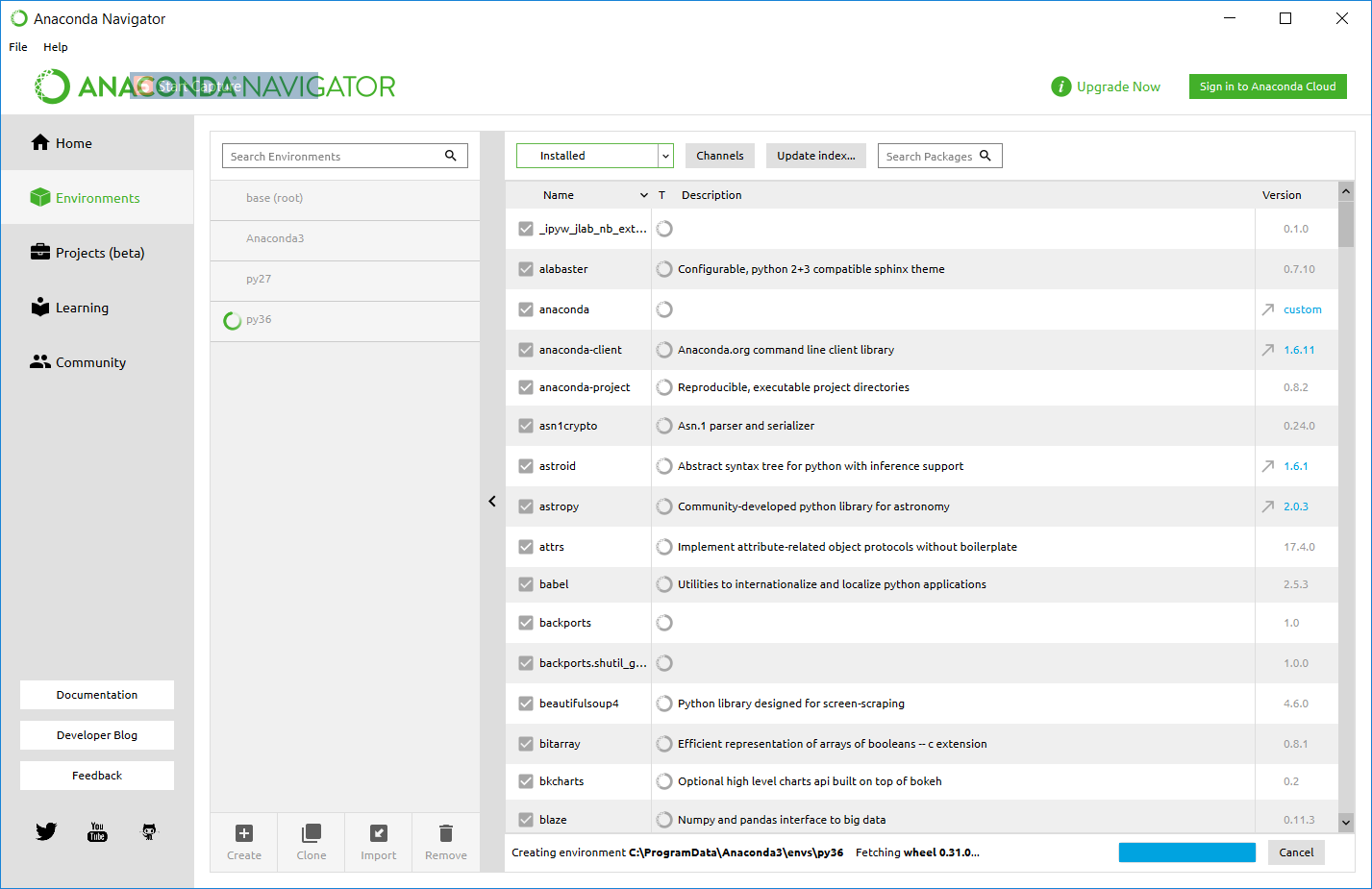
Suggestion from Web:



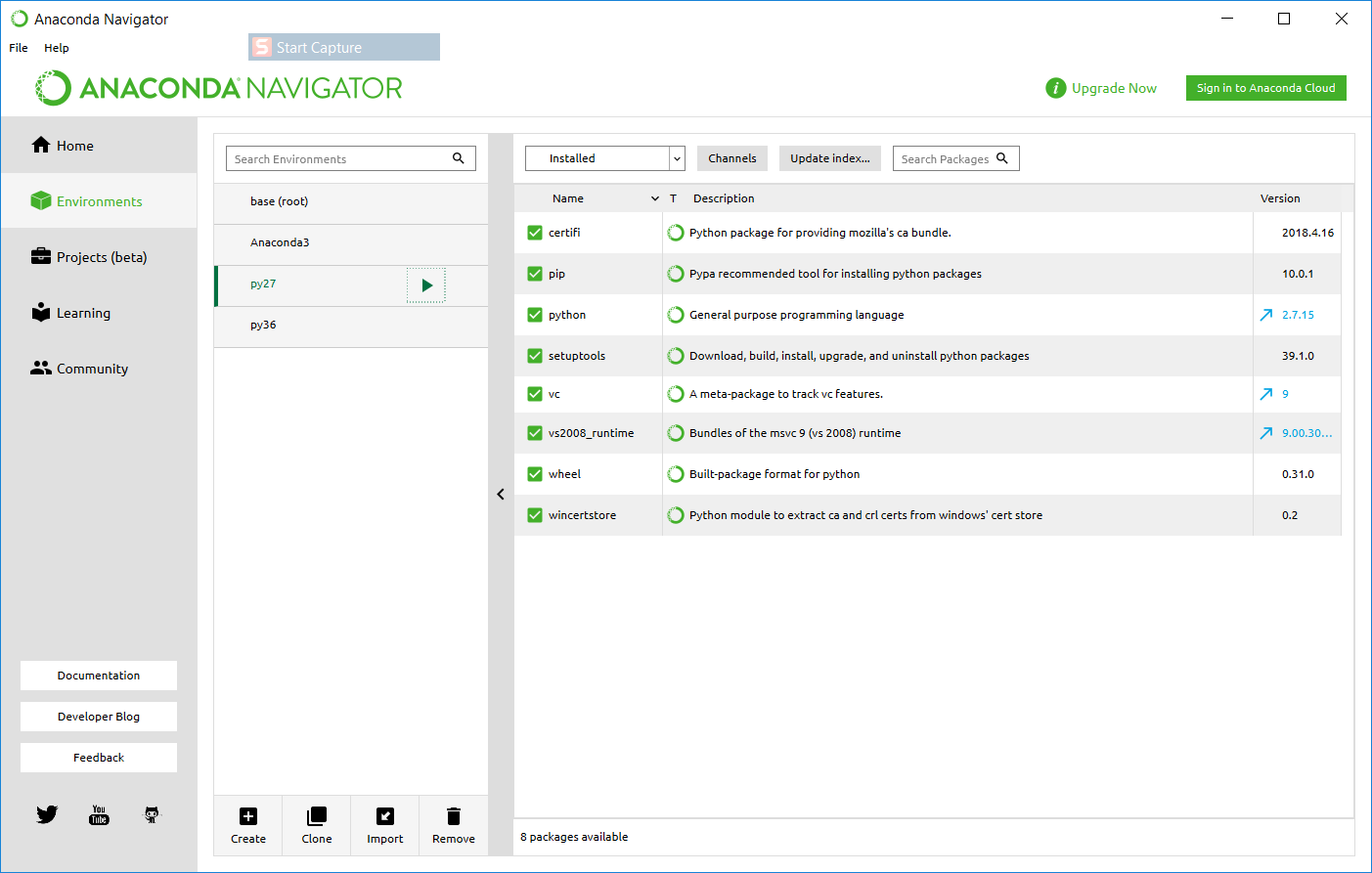
Create environment py36 (Python3.6)



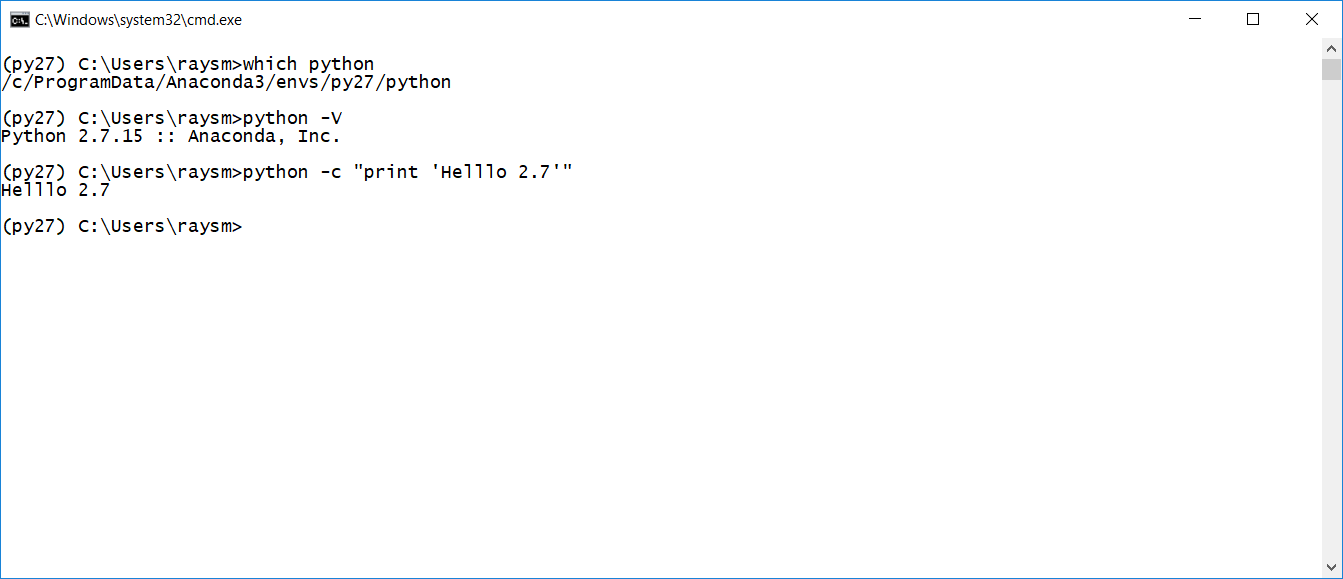
Creating py36 in progress



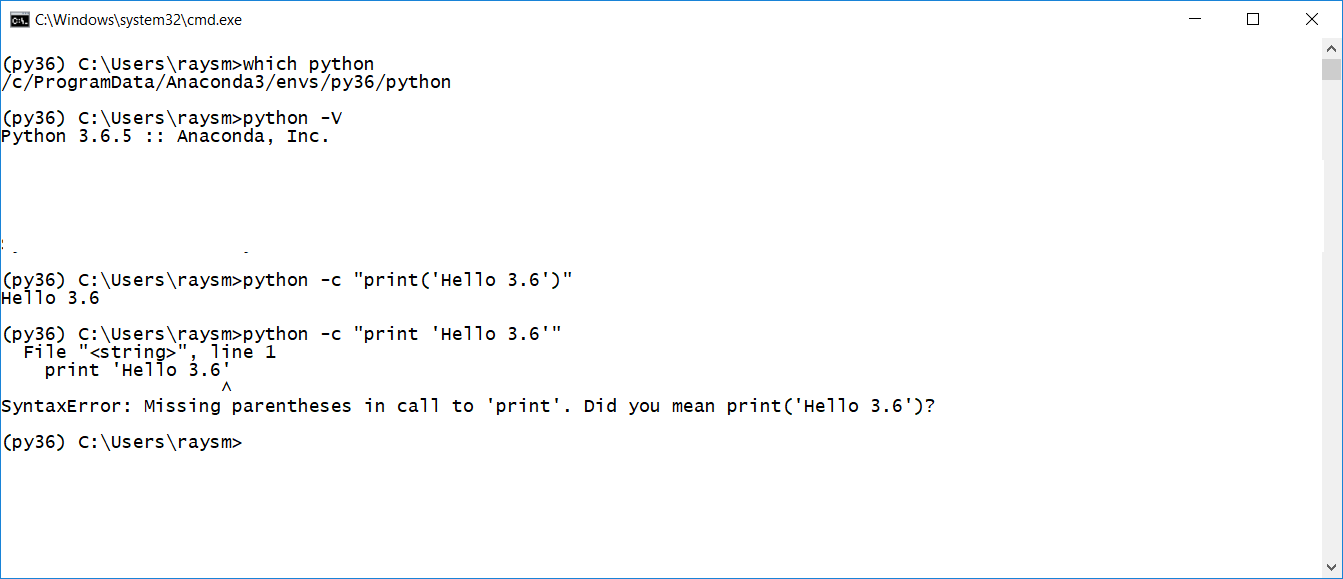
Selecting py27



Open Terminal



Open Terminal on py36



Getting / Using a Development Environment

1. Get Anaconda3 OR Anaconda2
   1. Install any additionally necessary modules
2. Get Eclipse
   1. Install Eclipse
   2. Get **PyDev** – Python Plugin (Maybe already installed)
      1. PyDev is a Python Development Environment (Python IDE plugin for Eclipse).
3. Connecting Eclipse to Anaconda

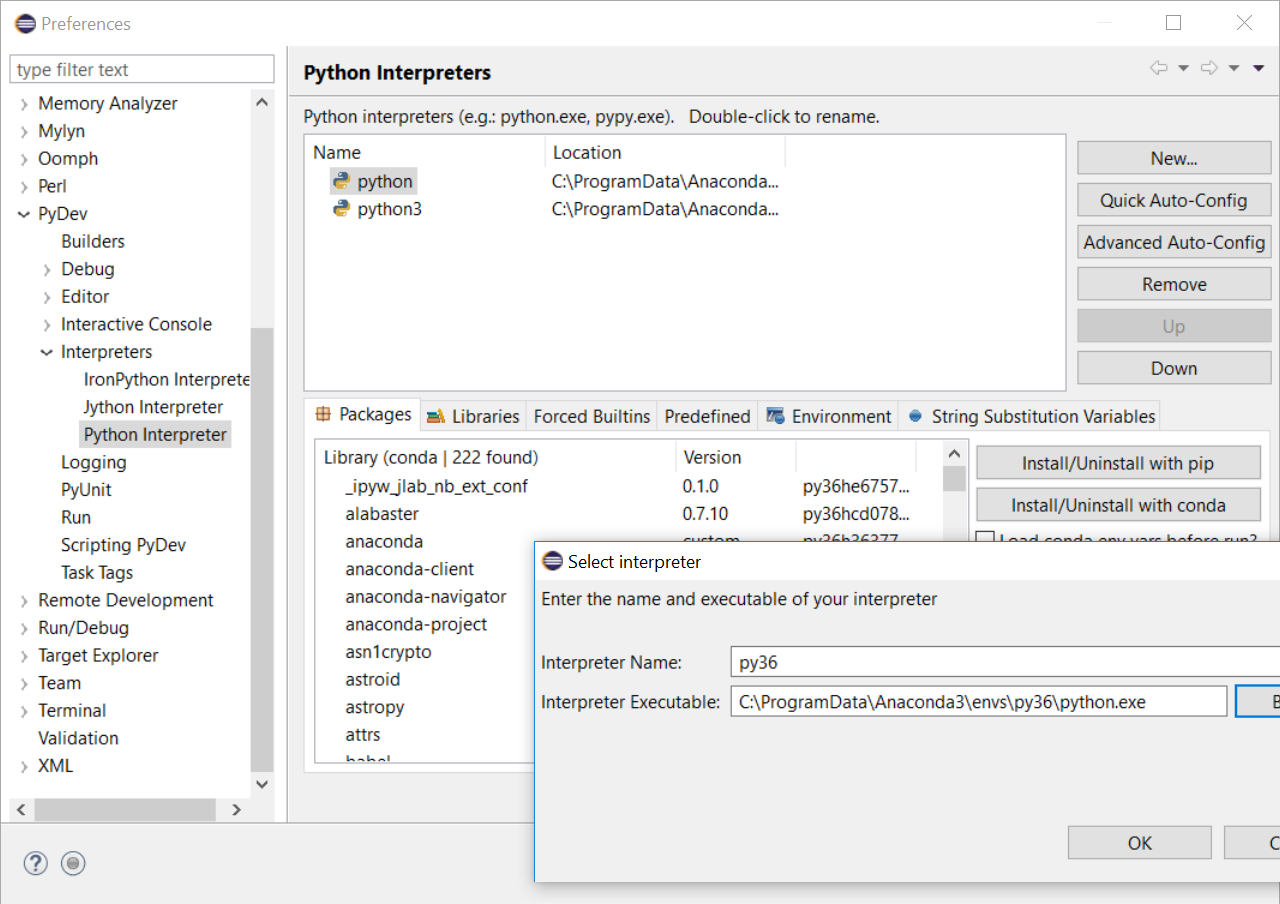
<https://github.com/mikec964/chelmbigstock/wiki/Set-up-Eclipse-(PyDev)-to-use-Anaconda-with-Python-2.7-and-3.x>

* 1. Add / Modify via Eclipse: Window 🡺 PyDev 🡺 Interpreters 🡺 Python Interpreter
     1. Python Interpreters: e.g.

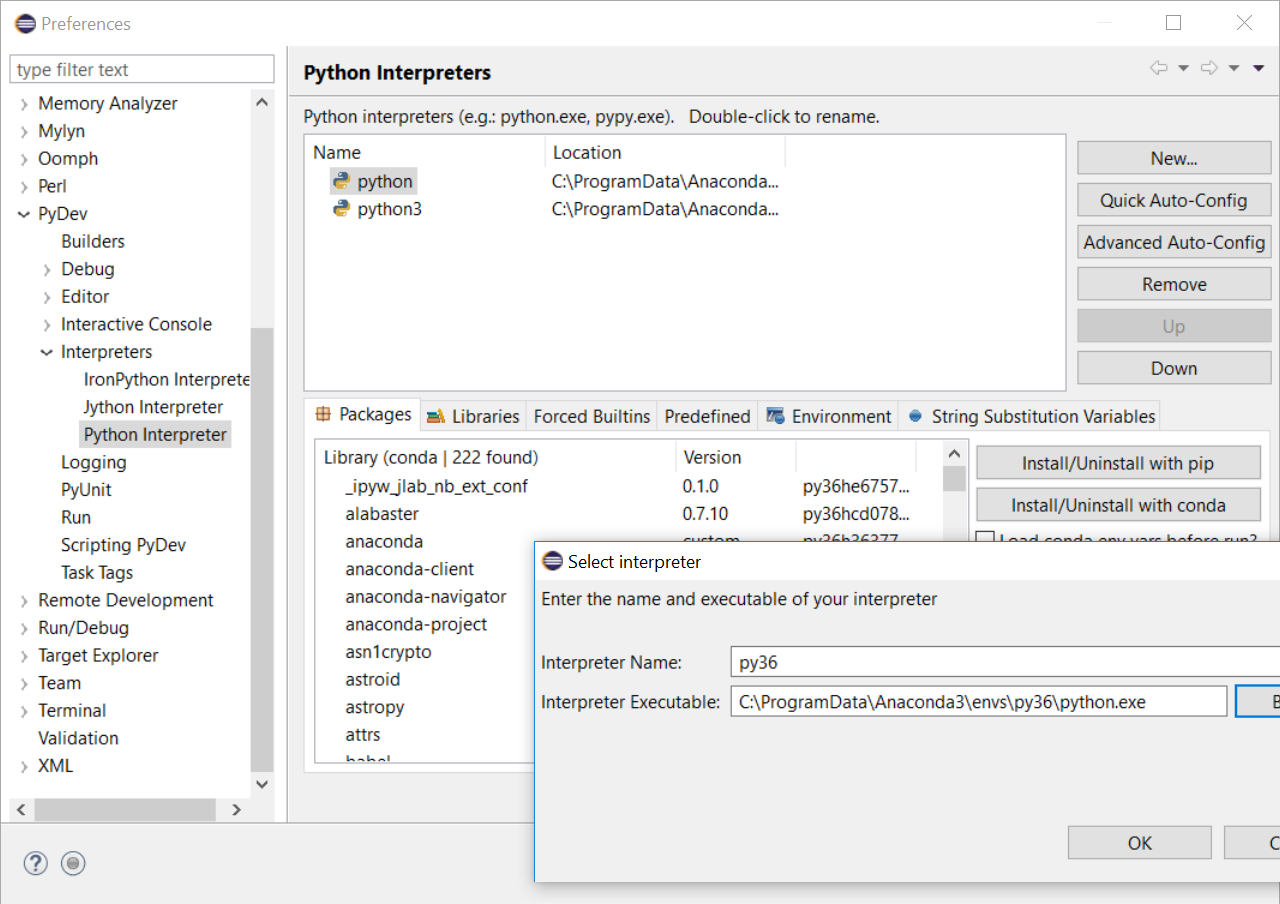
C:\ProgramData\Anaconda3\envs\py36\python.exe

* 1. Although Eclipse has its own installation mechanism, I’ve found using Anaconda’s easier to use

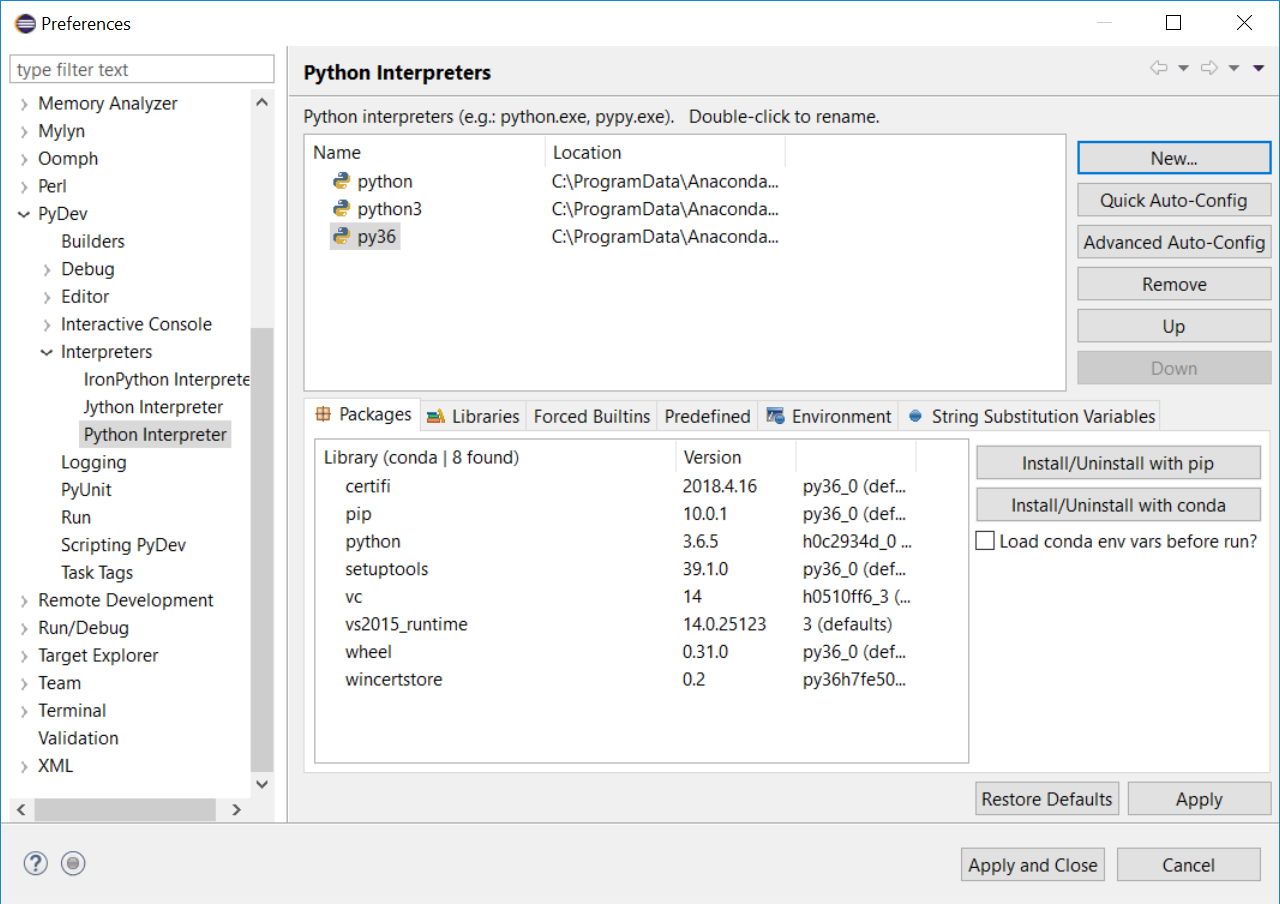
Connecting Eclipse to Anaconda



Continuing Eclipse to Anaconda connection



After Adding Interpreter



Note that Eclipse, finds the libraries, given the interpreter location