

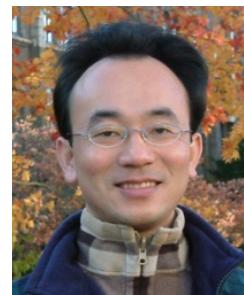
Social Computing and Big Data Analytics

社群運算與大數據分析



Big Data Analytics with Numpy in Python (Python Numpy 大數據分析)

1052SCBDA05
MIS MBA (M2226) (8606)
Wed, 8,9, (15:10-17:00) (L206)



Min-Yuh Day
戴敏育
Assistant Professor
專任助理教授

Dept. of Information Management, Tamkang University
淡江大學 資訊管理學系

<http://mail.tku.edu.tw/myday/>

2017-03-15



課程大綱 (Syllabus)

週次 (Week) 日期 (Date) 內容 (Subject/Topics)

- | | | |
|---|------------|--|
| 1 | 2017/02/15 | Course Orientation for Social Computing and
Big Data Analytics
(社群運算與大數據分析課程介紹) |
| 2 | 2017/02/22 | Data Science and Big Data Analytics:
Discovering, Analyzing, Visualizing and Presenting Data
(資料科學與大數據分析：
探索、分析、視覺化與呈現資料) |
| 3 | 2017/03/01 | Fundamental Big Data: MapReduce Paradigm,
Hadoop and Spark Ecosystem
(大數據基礎：MapReduce典範、
Hadoop與Spark生態系統) |

課程大綱 (Syllabus)

週次 (Week) 日期 (Date) 內容 (Subject/Topics)

- | | | |
|---|------------|---|
| 4 | 2017/03/08 | Big Data Processing Platforms with SMACK:
Spark, Mesos, Akka, Cassandra and Kafka
(大數據處理平台SMACK：
Spark, Mesos, Akka, Cassandra, Kafka) |
| 5 | 2017/03/15 | Big Data Analytics with Numpy in Python
(Python Numpy 大數據分析) |
| 6 | 2017/03/22 | Finance Big Data Analytics with Pandas in Python
(Python Pandas 財務大數據分析) |
| 7 | 2017/03/29 | Text Mining Techniques and
Natural Language Processing
(文字探勘分析技術與自然語言處理) |
| 8 | 2017/04/05 | Off-campus study (教學行政觀摩日) |

課程大綱 (Syllabus)

週次 (Week) 日期 (Date) 內容 (Subject/Topics)

9 2017/04/12 Social Media Marketing Analytics
(社群媒體行銷分析)

10 2017/04/19 期中報告 (Midterm Project Report)

11 2017/04/26 Deep Learning with Theano and Keras in Python
(Python Theano 和 Keras 深度學習)

12 2017/05/03 Deep Learning with Google TensorFlow
(Google TensorFlow 深度學習)

13 2017/05/10 Sentiment Analysis on Social Media with
Deep Learning
(深度學習社群媒體情感分析)

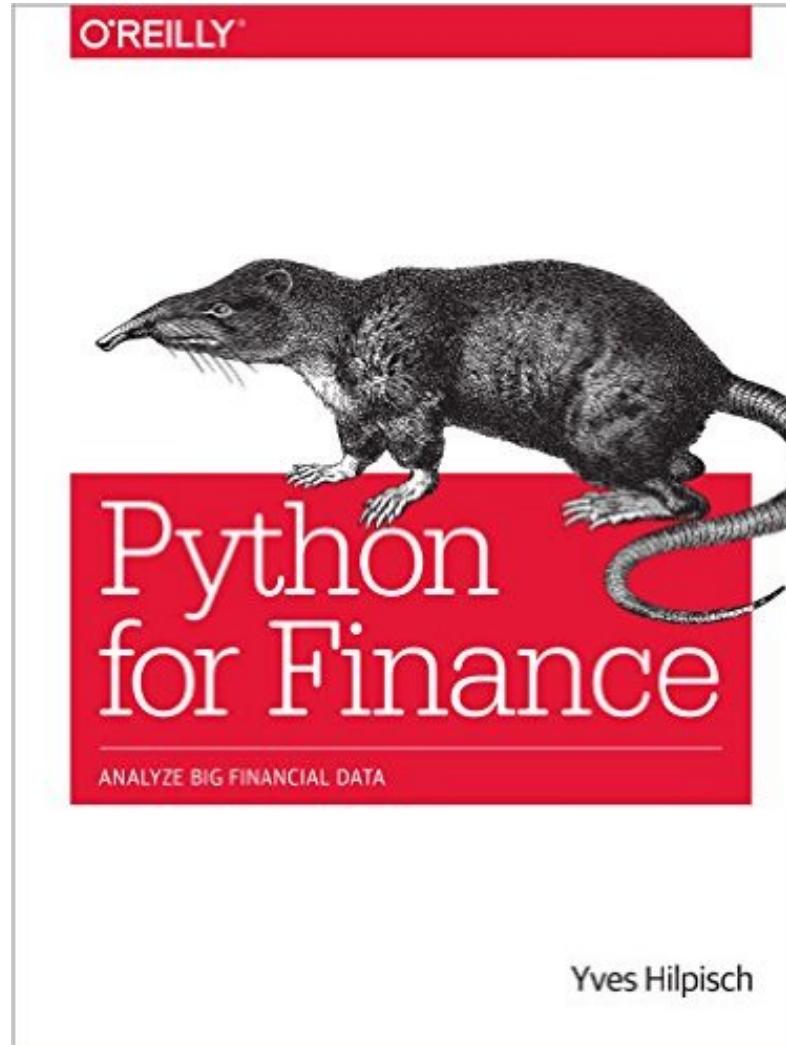
課程大綱 (Syllabus)

週次 (Week) 日期 (Date) 內容 (Subject/Topics)

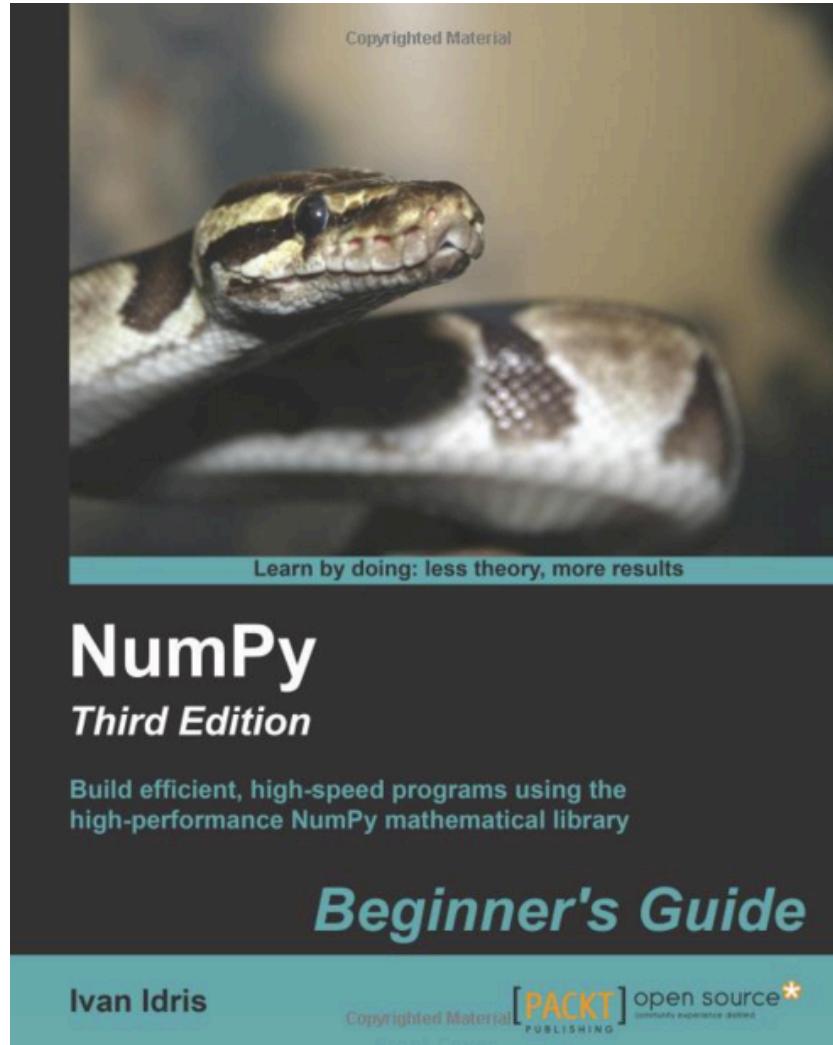
- | | | |
|----|------------|--|
| 14 | 2017/05/17 | Social Network Analysis (社會網絡分析) |
| 15 | 2017/05/24 | Measurements of Social Network (社會網絡量測) |
| 16 | 2017/05/31 | Tools of Social Network Analysis
(社會網絡分析工具) |
| 17 | 2017/06/07 | Final Project Presentation I (期末報告 I) |
| 18 | 2017/06/14 | Final Project Presentation II (期末報告 II) |



**Yves Hilpisch,
Python for Finance: Analyze Big Financial Data,
O'Reilly, 2014**

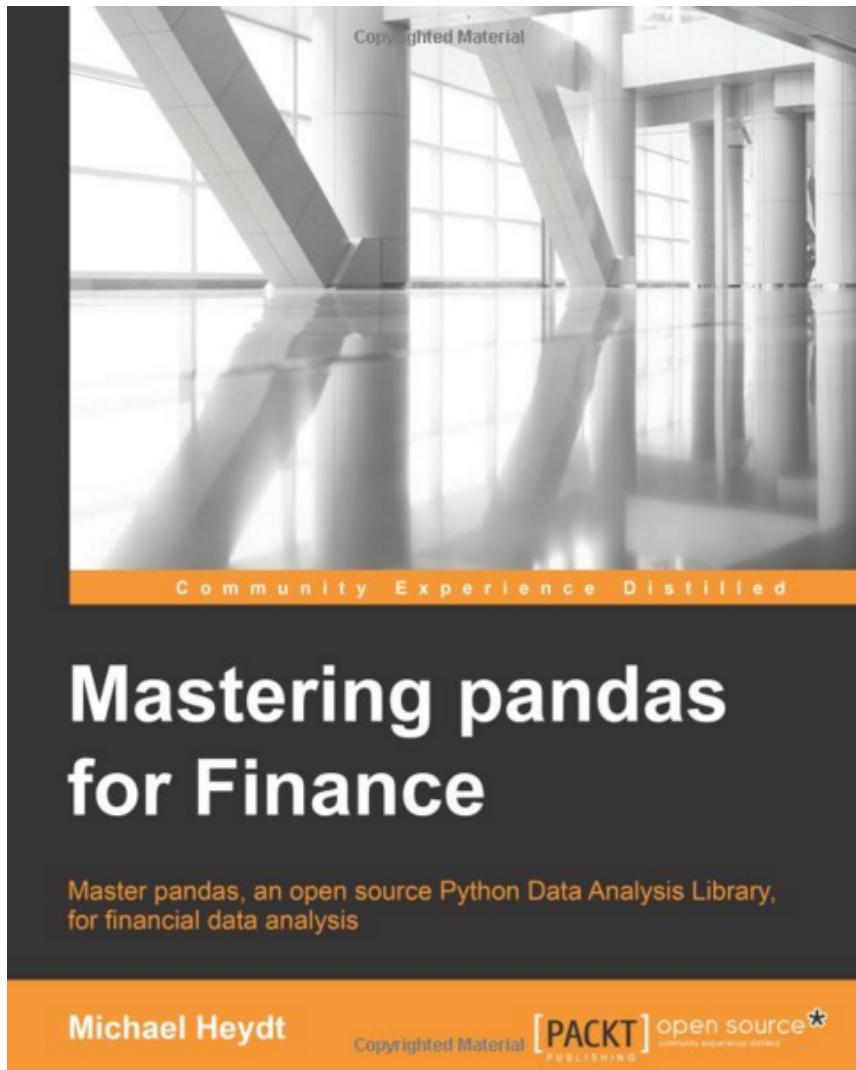


Ivan Idris,
Numpy Beginner's Guide, Third Edition
Packt Publishing, 2015



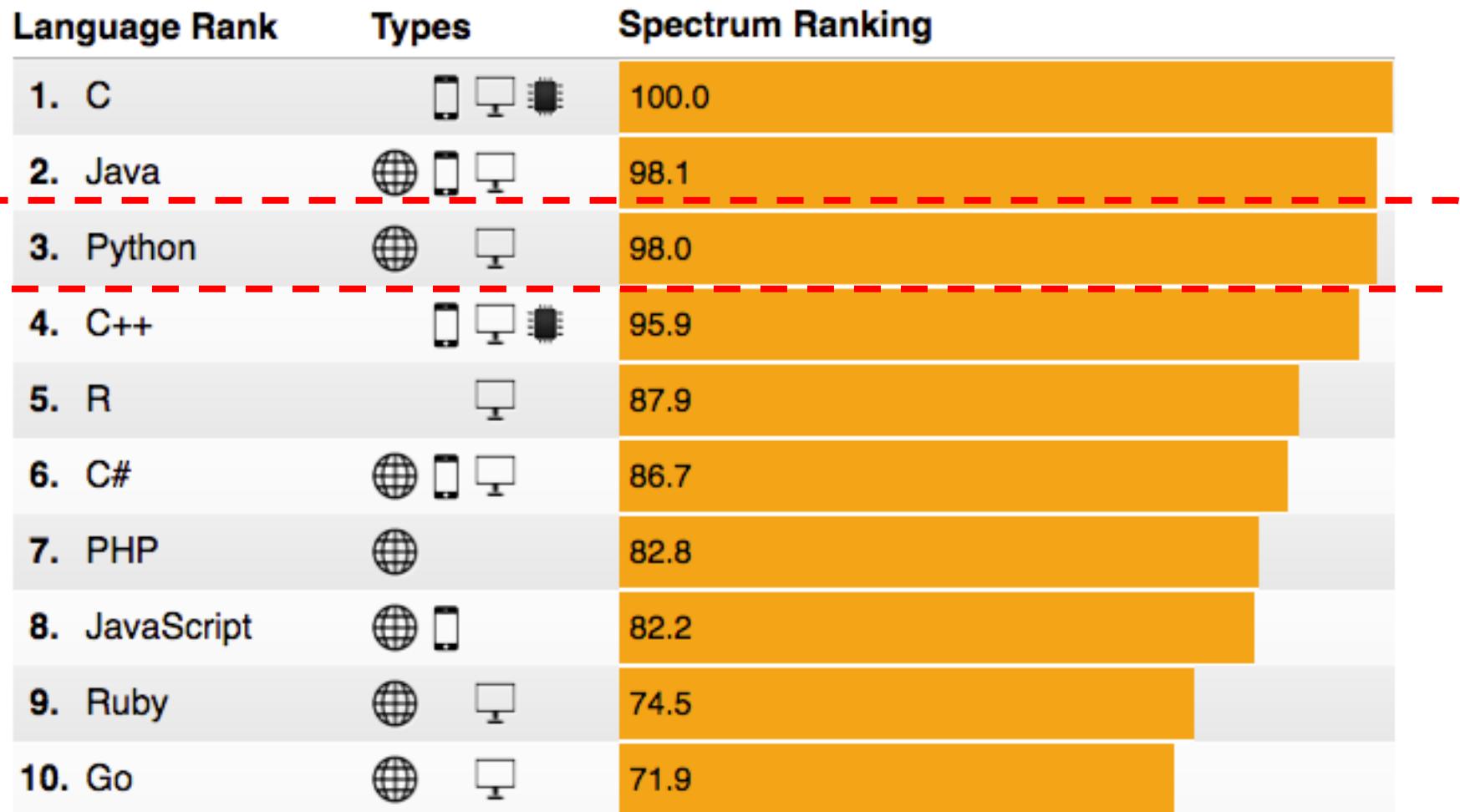
Source: <http://www.amazon.com/Numpy-Beginners-Guide-Ivan-Idris/dp/1785281968>

Michael Heydt , Mastering Pandas for Finance, Packt Publishing, 2015

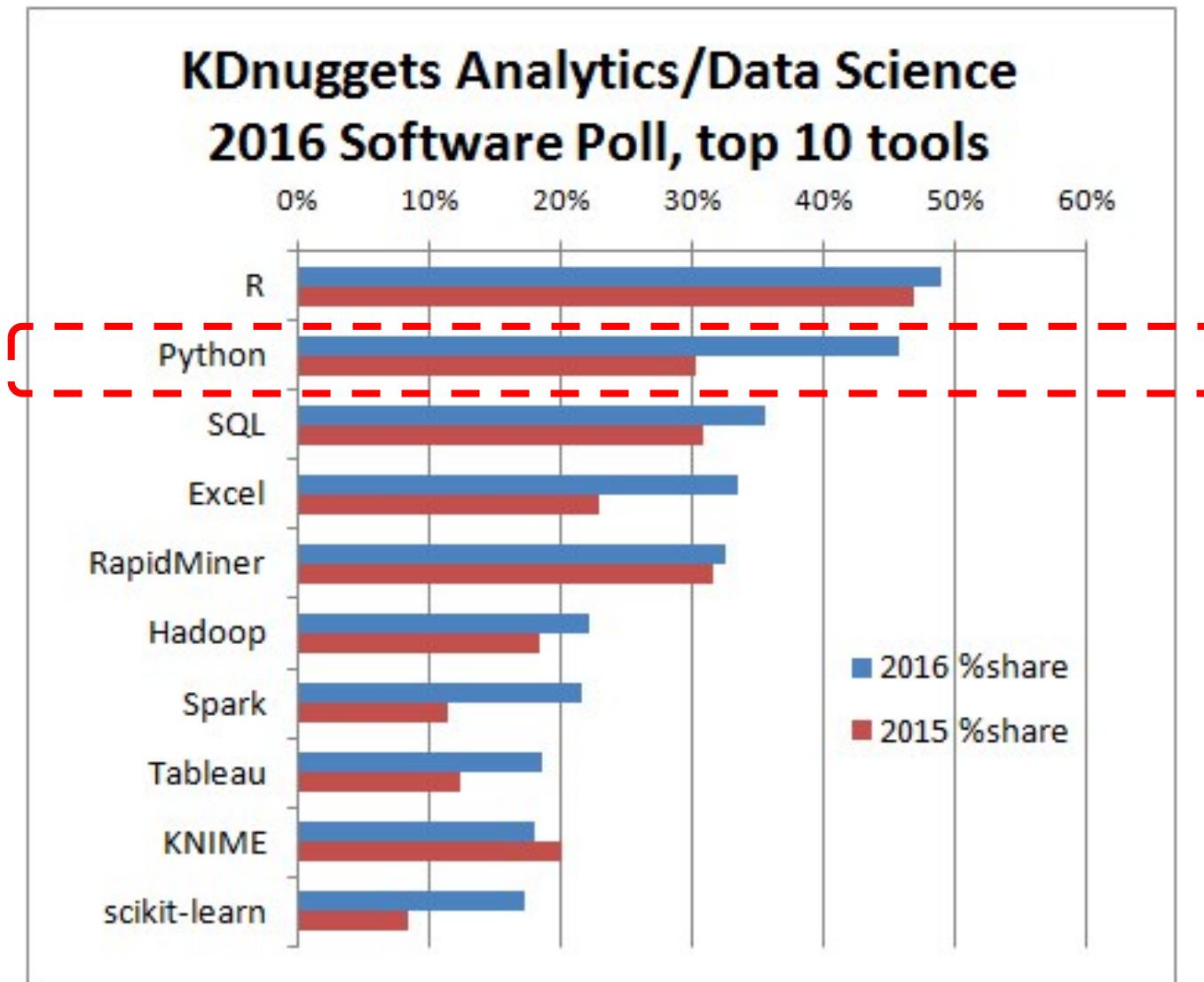


Source: <http://www.amazon.com/Mastering-Pandas-Finance-Michael-Heydt/dp/1783985100>

Python for Big Data Analytics



Python: Analytics and Data Science Software



Python

Python

PSF

Docs

PyPI

Jobs

Community



Search

GO

Socialize

Sign In

About

Downloads

Documentation

Community

Success Stories

News

Events

```
# Python 3: Simple output (with Unicode)
>>> print("Hello, I'm Python!")
Hello, I'm Python!

# Input, assignment
>>> name = input('What is your name?\n')
>>> print('Hi, %s.' % name)
What is your name?
Python
Hi, Python.
```



Quick & Easy to Learn

Experienced programmers in any other language can pick up Python very quickly, and beginners find the clean syntax and indentation structure easy to learn. [Whet your appetite](#) with our Python 3 overview.

1 2 3 4 5

Python is a programming language that lets you work quickly and integrate systems more effectively. [» Learn More](#)

Get Started

Download

Docs

Jobs

**Python is an
interpreted,
object-oriented,
high-level
programming language
with
dynamic semantics.**

NumPy



NumPy

Scipy.org

NumPy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

NumPy is licensed under the [BSD license](#), enabling reuse with few restrictions.

Getting Started

- [Getting NumPy](#)
- [Installing the SciPy Stack](#)
- [NumPy and SciPy documentation page](#)
- [NumPy Tutorial](#)
- [NumPy for MATLAB® Users](#)
- [NumPy functions by category](#)
- [NumPy Mailing List](#)

For more information on the SciPy Stack (for which NumPy provides the fundamental array data structure), see [scipy.org](#).

About NumPy

License

Old array packages

NumPy
is the
fundamental package
for
scientific computing
with Python.

Python versions (py2 and py3)

- Python 0.9.0 released in 1991 (first release)
- Python 1.0 released in 1994
- Python 2.0 released in 2000
- Python 2.6 released in 2008
- **Python 2.7 released in 2010**
- **Python 3.0 released in 2008**
- Python 3.3 released in 2010
- Python 3.4 released in 2014
- **Python 3.5 released in 2015**
- **Python 3.6 released in 2016**

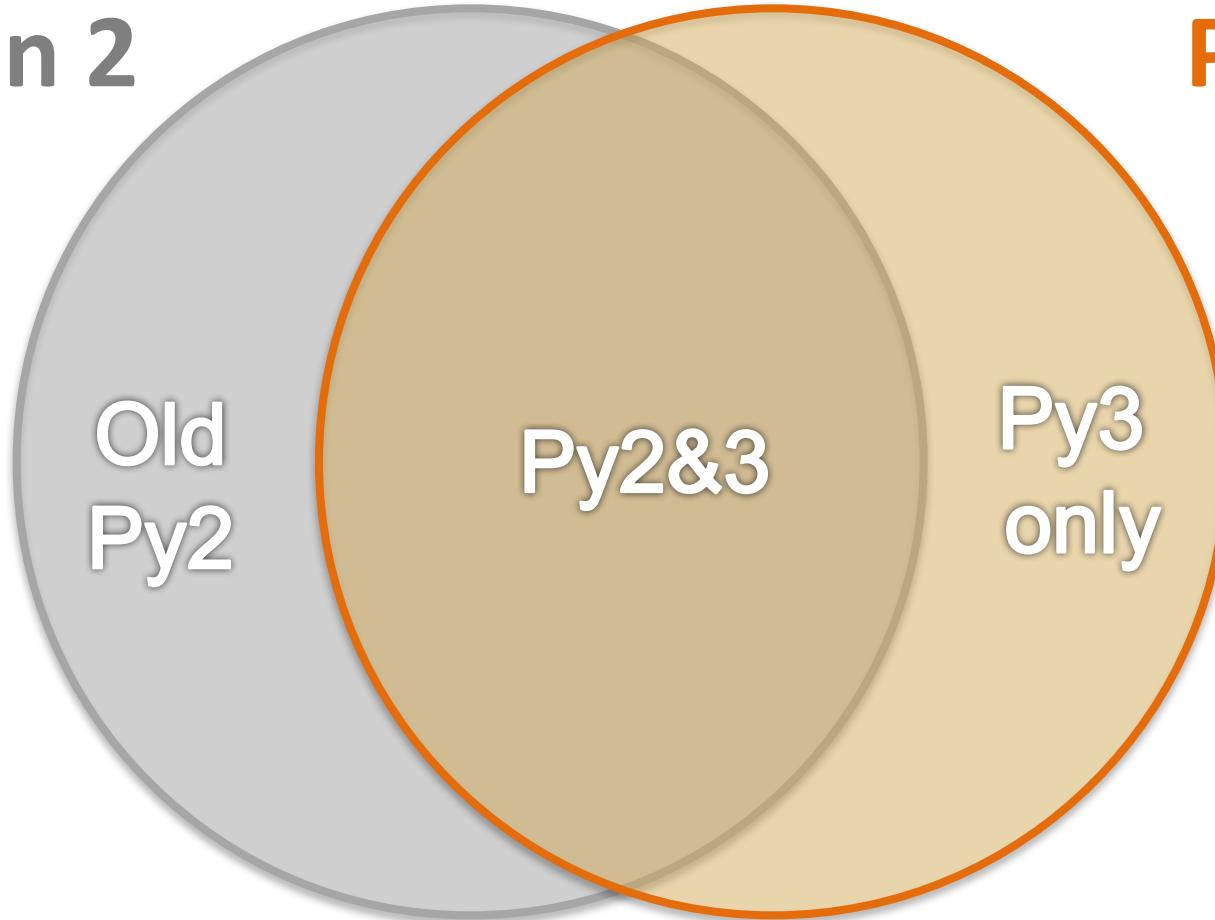
Python (Python 2.7 & Python 3.6)



Standard Syntax

Python 2

Python 3



Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield

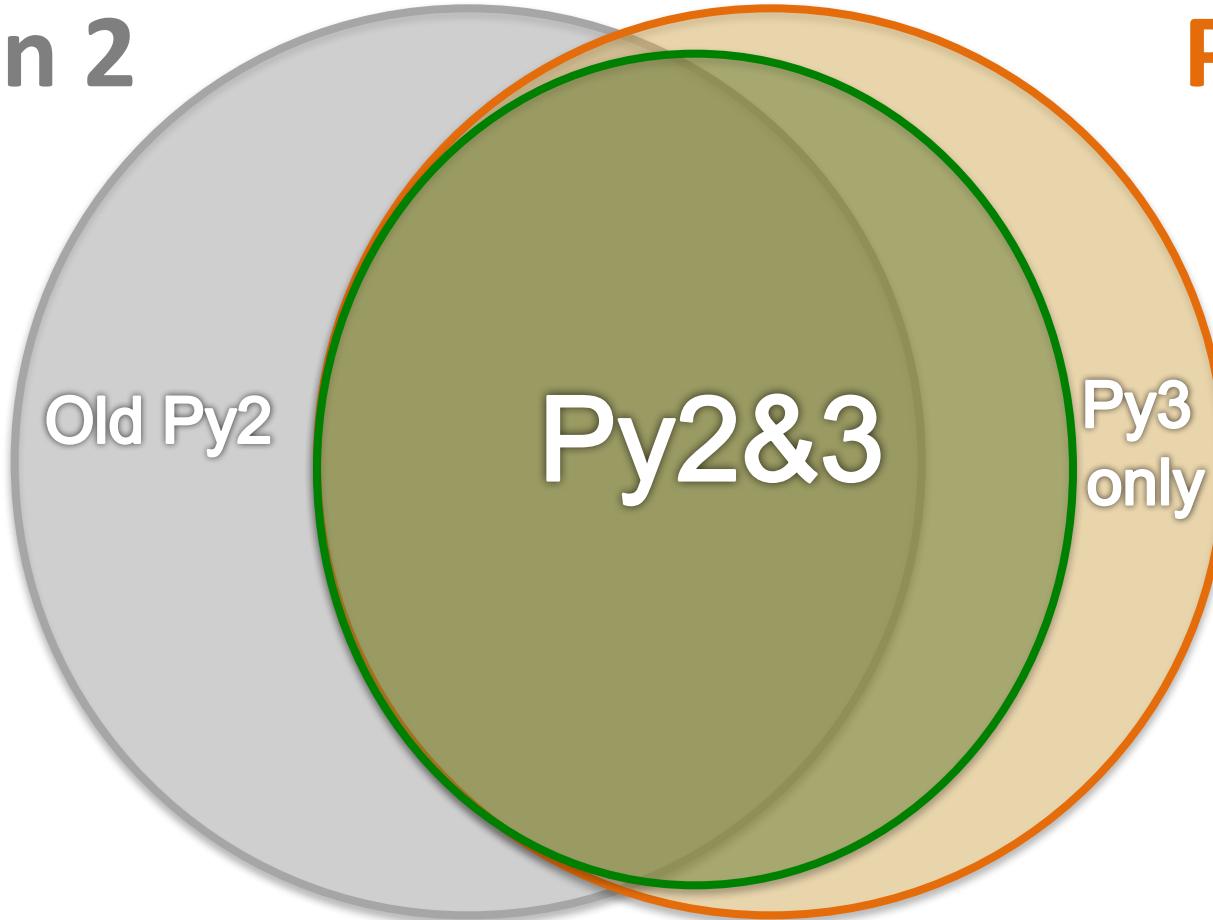
<https://www.youtube.com/watch?v=KOqk8j11aAI>

```
from __future__ import ...
```



Python 2

Python 3



Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield

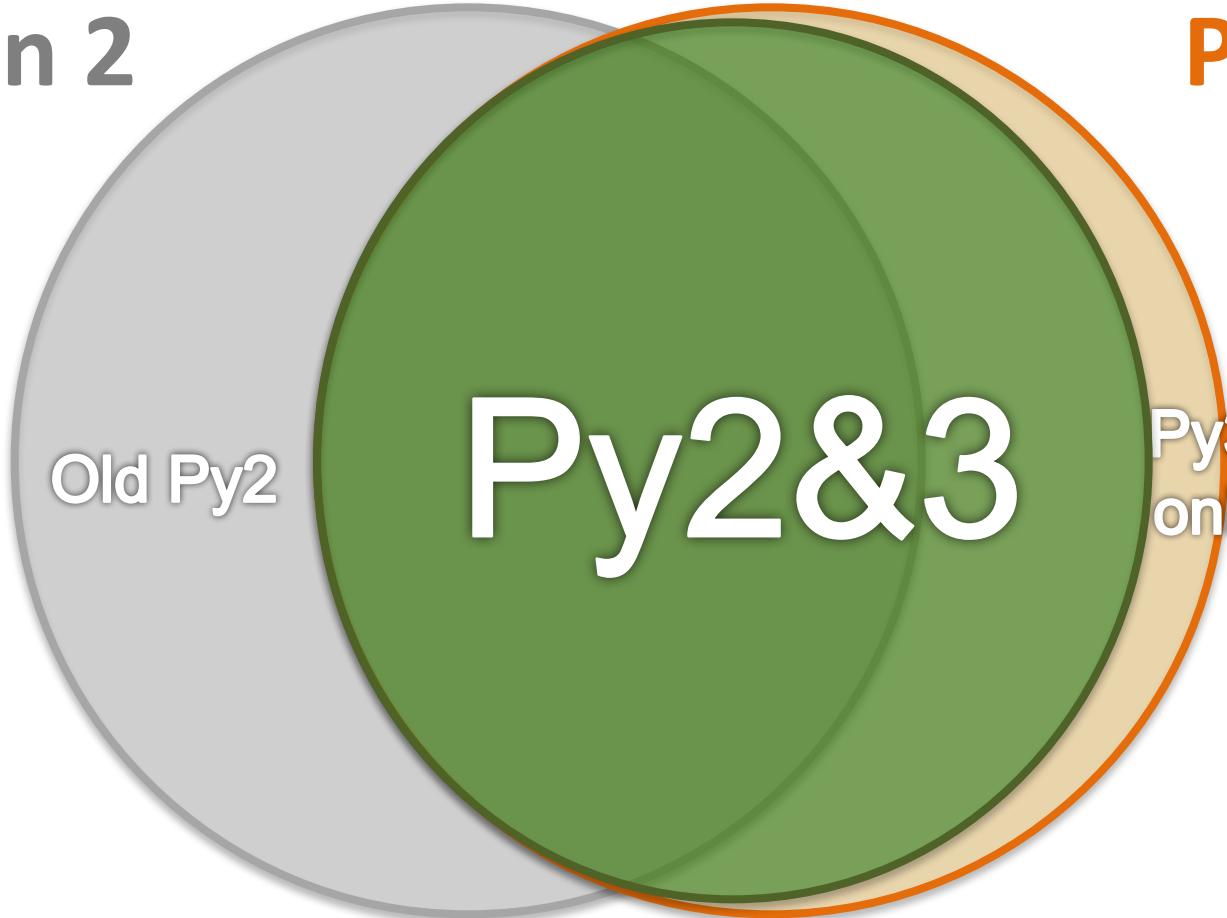
<https://www.youtube.com/watch?v=KOqk8j11aAI>

```
from future.builtins import *
```



Python 2

Python 3



Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield

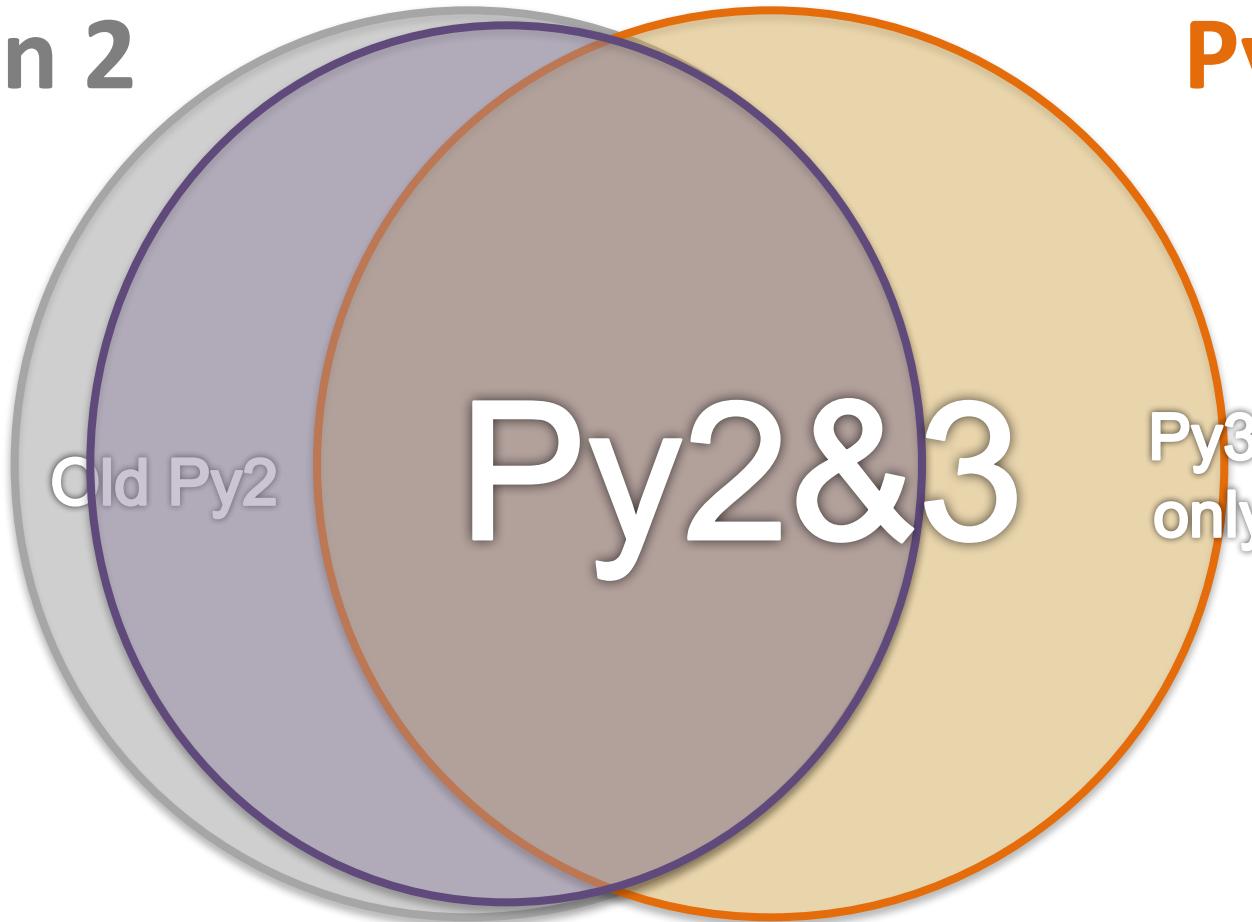
<https://www.youtube.com/watch?v=KOqk8j11aAI>

```
from past.builtins import *
```



Python 2

Python 3



Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield

<https://www.youtube.com/watch?v=KOqk8j11aAI>



**Leading Open
Data Science
Platform
Powered by
Python**

Anaconda



[Log In](#) [Get Support](#) [Search](#) [Contact](#)

[PRODUCTS](#) [COMMUNITY](#) [CONSULTING](#) [TRAINING](#) [ABOUT](#) [RESOURCES](#)

SUPERPOWERS FOR PEOPLE WHO CHANGE THE WORLD

Leading Open Data Science Platform Powered by Python

[DOWNLOAD ANACONDA](#)



ACCELERATE

Time-to-Value



CONNECT

Data, Analytics & Compute



EMPOWER

Data Science Teams



<https://www.continuum.io/>

Download Anaconda



[Log In](#) [Get Support](#) [Search](#) [Contact](#)

[PRODUCTS](#) [COMMUNITY](#) [CONSULTING](#) [TRAINING](#) [ABOUT](#) [RESOURCES](#)

DOWNLOAD ANACONDA NOW

Download for



GET SUPERPOWERS WITH ANACONDA

Anaconda is the leading open data science platform powered by Python. The open source version of Anaconda is a high performance distribution of Python and R and includes over 100 of the most popular Python, R and Scala packages for data science.

Which version should I download and install?

With Anaconda you can run multiple versions of Python in isolated environments, so choose the download with the Python version that you use more often, as that will be your default Python version.

<https://www.continuum.io/downloads>

Download Anaconda Python 3.6

Download for Windows

Download for macOS

Download for Linux

Anaconda 4.3.1

For macOS

macOS 10.12.2 users: To prevent permissions problems, we recommend that you upgrade to macOS 10.12.3 or later before installing Anaconda.

Anaconda is BSD licensed which gives you permission to use Anaconda commercially and for redistribution.

Changelog

Graphical Installer

1. Download the graphical installer
2. Double-click the downloaded .pkg file and follow the instructions

Command Line Installer

1. Download the command-line installer
2. Optional: Verify data integrity with [MD5](#) or [SHA-256](#) [More info](#)
3. In your terminal window type one of the below and follow the instructions:
Python 3.6 version

Python 3.6 version

GRAPHICAL INSTALLER (424M)

COMMAND-LINE INSTALLER (363M)

64-Bit

Python 2.7 version

GRAPHICAL INSTALLER (419M)

COMMAND-LINE INSTALLER (358M)

64-Bit

GET ANACONDA SUPPORT

OS X Anaconda Python 3.6

Installation

Command Line Installer

Download the command-line installer

In your terminal window type one of the below
and follow the instructions:

Python 3.6 version

```
bash Anaconda3-4.3.1-MacOSX-x86_64.sh
```

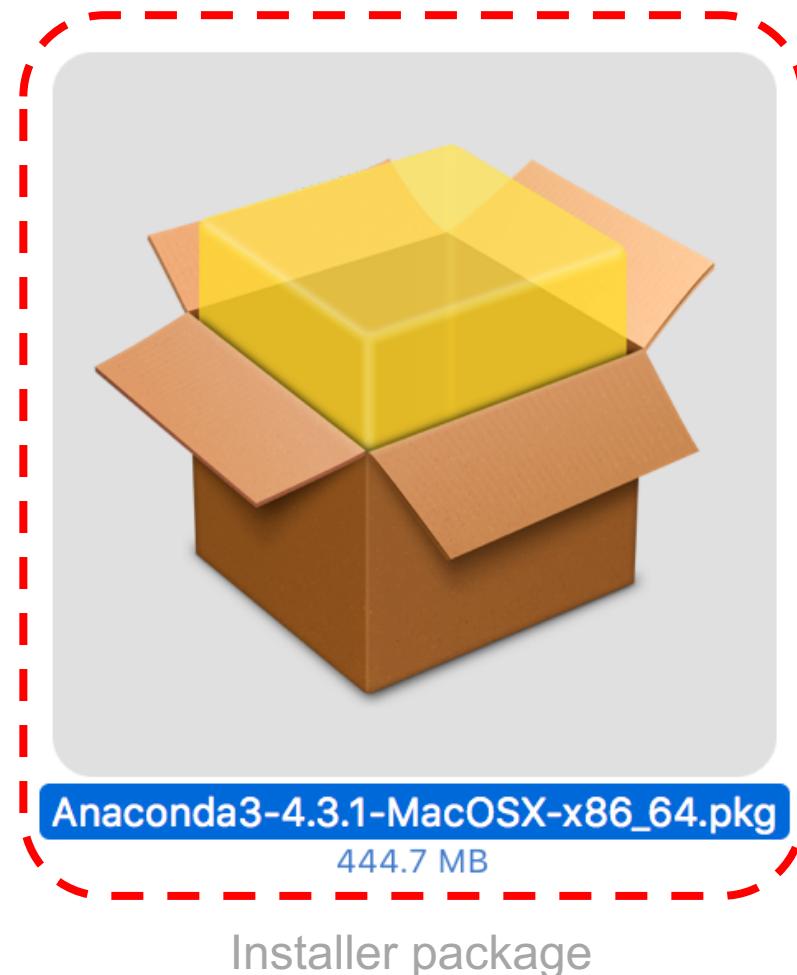
Python 2.7 version

```
bash Anaconda2-4.3.1-MacOSX-x86_64.sh
```

OS X Anaconda 3 - 4.3.1

Python 3.6 Installation

Anaconda3-4.3.1-MacOSX-x86_64.pkg



Install Anaconda 3

Install Anaconda3

Welcome to the Anaconda3 Installer

- **Introduction**
- Read Me
- License
- Destination Select
- Installation Type
- Installation
- Summary

You will be guided through the steps necessary to install this software.

 ANACONDA®

Go Back Continue

Install Anaconda 3

Install Anaconda3

Important Information

- Introduction
- Read Me**
- License
- Destination Select
- Installation Type
- Installation
- Summary

 **ANACONDA®**

Anaconda is a modern open source analytics platform powered by Python. See <https://www.continuum.io/downloads/>.

By default, this installer modifies your bash profile to put Anaconda in your PATH. To disable this, choose "Customize" at the "Installation Type" phase, and disable the "Modify PATH" option. If you do not do this, you will need to add `~/anaconda/bin` to your PATH manually to run the commands, or run all anaconda commands explicitly from that path.

To install to a different location, select "Change Install Location..." at the "Installation Type" phase, the choose "Install on a specific disk...", choose the disk you wish to install on, and click "Choose Folder...". The "Install for me only" option will install anaconda to the default location, `~/anaconda`.

The packages included in this installation are:

- alabaster 0.7.9

Print...

Save...

Go Back

Continue

28

Install Anaconda 3

Install Anaconda3

Software License Agreement

=====

Anaconda License

=====

Copyright 2016, Continuum Analytics, Inc.

All rights reserved under the 3-clause BSD License:

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of Continuum Analytics, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS



ANACONDA

Install Anaconda 3

Install Anaconda3

Software License Agreement

=====

Anaconda License

=====

Copyright 2016, Continuum Analytics, Inc.

To continue installing the software you must agree to the terms of the software license agreement.

Click Agree to continue or click Disagree to cancel the installation and quit the Installer.

Read License Disagree Agree

* Neither the name of Continuum Analytics, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS

Print... Save... Go Back Continue



ANACONDA®

Install Anaconda 3

Install Anaconda3

Standard Install on "Macintosh HD"

- Introduction
- Read Me
- License
- Destination Select
- Installation Type**
- Installation
- Summary

This will take 1.4 GB of space on your computer.

Click Install to perform a standard installation of this software in your home folder. Only the current user of this computer will be able to use this software.

Change Install Location...

Customize Go Back Install



ANACONDA®

Install Anaconda 3

Install Anaconda3

Select a Destination

- Introduction
- Read Me
- License
- **Destination Select**
- Installation Type
- Installation
- Summary

How do you want to install this software?

 Install for all users of this computer

 **Install for me only**

 Install on a specific disk...

Installing this software requires 1.4 GB of space.
You have chosen to install this software in your home folder.
Only the current user will be able to use this software.

Go Back Continue



Install Anaconda 3

Install Anaconda3

Standard Install on "Macintosh HD"

- Introduction
- Read Me
- License
- Destination Select
- Installation Type**
- Installation
- Summary

This will take 1.4 GB of space on your computer.

Click Install to perform a standard installation of this software in your home folder. Only the current user of this computer will be able to use this software.

Change Install Location...

Customize Go Back **Install**



ANACONDA®

Install Anaconda 3

Install Anaconda3

Installing Anaconda3

- Introduction
- Read Me
- License
- Destination Select
- Installation Type
- **Installation**
- Summary

Registering updated applications...

Install time remaining: About a minute

Go Back Continue



ANACONDA®

Install Anaconda 3

The installation was completed successfully.

Anaconda is the leading open data science platform powered by Python.

Share your notebooks and packages on Anaconda Cloud!
[Sign up for free](#)

178 python packages included.

**Supported packages:
453**



ANACONDA®

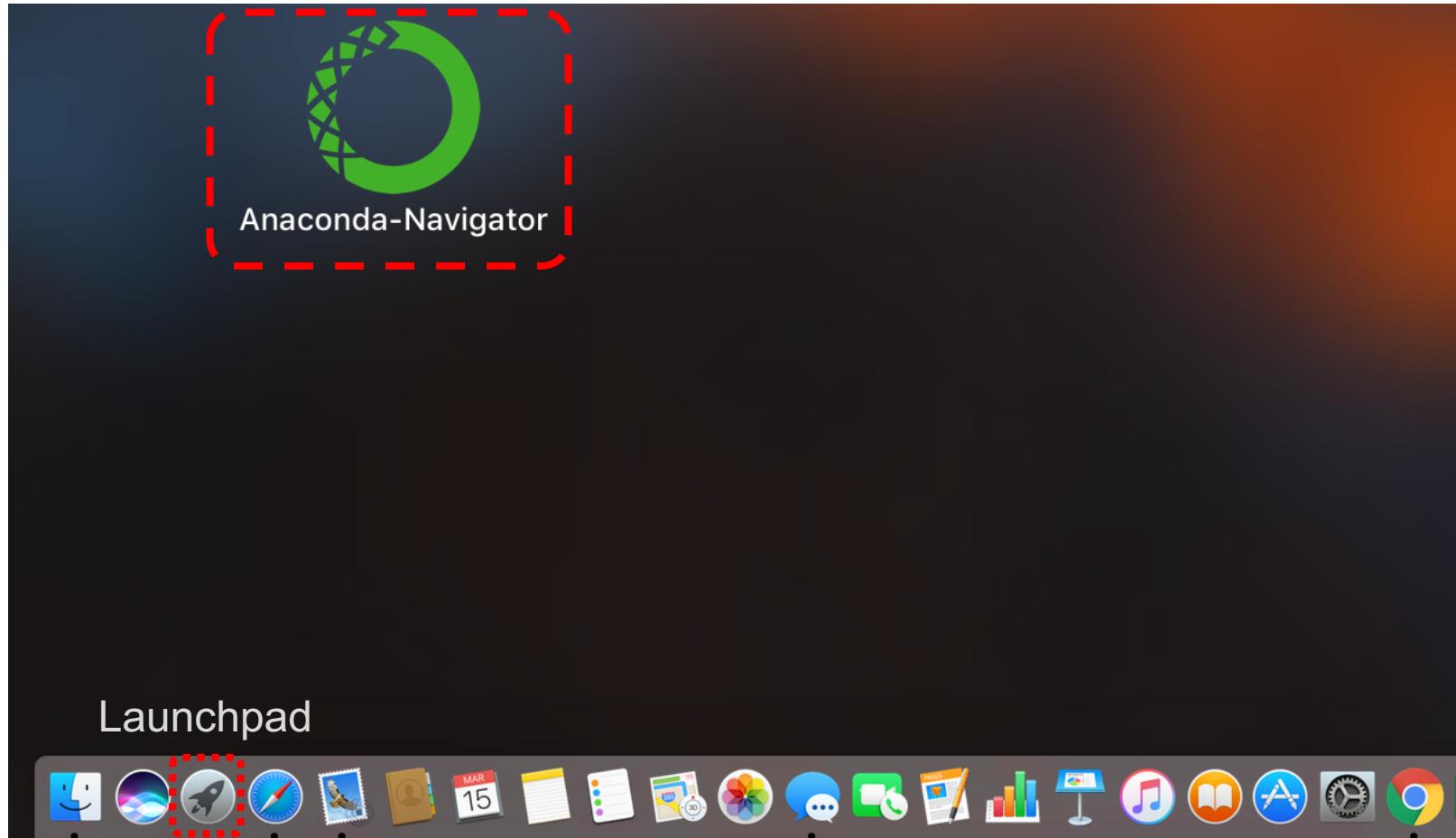
Go Back Close

Install Anaconda 3

1	_license	1.1	51	heapdict	1.0.0	101	partd	0.3.7	151	sip	4.18	py36_0
2	alabaster	0.7.9	52	icu	54.1	102	path.py	10.0	152	six	1.10.0	py36_0
3	anaconda	4.3.1	53	idna	2.2	103	pathlib2	2.2.0	153	snowballstemmer	1.2.1	py36_0
4	anaconda-client	1.6.0	54	imagesize	0.7.1	104	patsy	0.4.1	154	sockjs-tornado	1.0.3	py36_0
5	anaconda-navigator	1.5.0	55	ipykernel	4.5.2	105	pep8	1.7.0	155	sphinx	1.5.1	py36_0
6	anaconda-project	0.4.1	56	ipython	5.1.0	106	pexpect	4.2.1	156	spyder	3.1.2	py36_0
7	appnope	0.1.0	57	ipython_genutils	0.1.0	107	pickleshare	0.7.4	157	sqlalchemy	1.1.5	py36_0
8	appscript	1.0.1	58	ipywidgets	5.2.2	108	pillow	4.0.0	158	sqlite	3.13.0	0
9	astroid	1.4.9	59	isort	4.2.5	109	pip	9.0.1	159	statsmodels	0.6.1	np111py36_1
10	astropy	1.3	60	itsdangerous	0.24	110	ply	3.9	160	sympy	1.0	py36_0
11	babel	2.3.4	61	jbig	2.1	111	prompt_toolkit	1.0.9	161	terminado	0.6	py36_0
12	backports	1.0	62	jdcal	1.3	112	psutil	5.0.1	162	tk	8.5.18	0
13	beautifulsoup4	4.5.3	63	jedi	0.9.0	113	ptyprocess	0.5.1	163	toolz	0.8.2	py36_0
14	bitarray	0.8.1	64	jinja2	2.9.4	114	py	1.4.32	164	tornado	4.4.2	py36_0
15	blaze	0.10.1	65	jpeg	9b	115	pyasn1	0.1.9	165	traitlets	4.3.1	py36_0
16	bokeh	0.12.4	66	jsonschema	2.5.1	116	pycosat	0.6.1	166	unicodecsv	0.14.1	py36_0
17	boto	2.45.0	67	jupyter	1.0.0	117	pycparser	2.17	167	wcwidth	0.1.7	py36_0
18	bottleneck	1.2.0	68	jupyter_client	4.4.0	118	pycrypto	2.6.1	168	werkzeug	0.11.15	py36_0
19	cffi	1.9.1	69	jupyter_console	5.0.0	119	pycurl	7.43.0	169	wheel	0.29.0	py36_0
20	chardet	2.3.0	70	jupyter_core	4.2.1	120	pyflakes	1.5.0	170	widgetsnbextension	1.2.6	py36_0
21	chest	0.2.3	71	lazy-object-proxy	1.2.2	121	pygments	2.1.3	171	wrapt	1.10.8	py36_0
22	click	6.7	72	libiconv	1.14	122	pylint	1.6.4	172	xlrd	1.0.0	py36_0
23	cloudpickle	0.2.2	73	libpng	1.6.27	123	pyopenssl	16.2.0	173	xlsxwriter	0.9.6	py36_0
24	clyent	1.2.2	74	libtiff	4.0.6	124	pyparsing	2.1.4	174	xlwings	0.10.2	py36_0
25	colorama	0.3.7	75	libxml2	2.9.4	125	pyqt	5.6.0	175	xlwt	1.2.0	py36_0
26	conda	4.3.14	76	libxslt	1.1.29	126	pytables	3.3.0	176	xz	5.2.2	1
27	conda-env	2.6.0	77	llVMLite	0.15.0	127	pytest	3.0.5	177	yaml	0.1.6	0
28	configobj	5.0.6	78	locket	0.2.0	128	python	3.6.0	178	zlib	1.2.8	3
29	contextlib2	0.5.4	79	lxml	3.7.2	129	python-dateutil	2.6.0		py36_0		
30	cryptography	1.7.1	80	markupsafe	0.23	130	python.app	1.2		py36_4		
31	curl	7.52.1	81	matplotlib	2.0.0	131	pytz	2016.10		py36_0		
32	cycler	0.10.0	82	mistune	0.7.3	132	pyyaml	3.12		py36_0		
33	cython	0.25.2	83	mkl	2017.0.1	133	pyzmq	16.0.2		py36_0		
34	cytoolz	0.8.2	84	mkl-service	1.1.2	134	qt	5.6.2		0		
35	dask	0.13.0	85	mpmath	0.19	135	qtawesome	0.4.3		py36_0		
36	datashape	0.5.4	86	multipledispatch	0.4.9	136	qtconsole	4.2.1		py36_1		
37	decorator	4.0.11	87	nbconvert	4.2.0	137	qtpy	1.2.1		py36_0		
38	dill	0.2.5	88	nbformat	4.2.0	138	readline	6.2		2		
39	docutils	0.13.1	89	networkx	1.11	139	redis	3.2.0		0		
40	entrypoints	0.2.2	90	nltk	3.2.2	140	redis-py	2.10.5		py36_0		
41	et_xmlfile	1.0.1	91	nose	1.3.7	141	requests	2.12.4		py36_0		
42	fastcache	1.0.2	92	notebook	4.3.1	142	rope	0.9.4		py36_1		
43	flask	0.12	93	numba	0.30.1	143	ruamel.yaml	0.11.14		py36_1		
44	flask-cors	3.0.2	94	numexpr	2.6.1	144	scikit-image	0.12.3		np111py36_1		
45	freetype	2.5.5	95	numpy	1.11.3	145	scikit-learn	0.18.1		np111py36_1		
46	get_terminal_size	1.0.0	96	numpydoc	0.6.0	146	scipy	0.18.1		np111py36_1		
47	gevent	1.2.1	97	odo	0.5.0	147	seaborn	0.7.1		py36_0		
48	greenlet	0.4.11	98	openpyxl	2.4.1	148	setuptools	27.2.0		py36_0		
49	h5py	2.6.0	99	openssl	1.0.2k	149	simplegeneric	0.8.1		py36_1		
50	hdf5	1.8.17	100	pandas	0.19.2	150	singledispatch	3.4.0.3		py36_0		

178
python
packages
included.

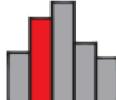
Anaconda-Navigator



Anaconda-Navigator

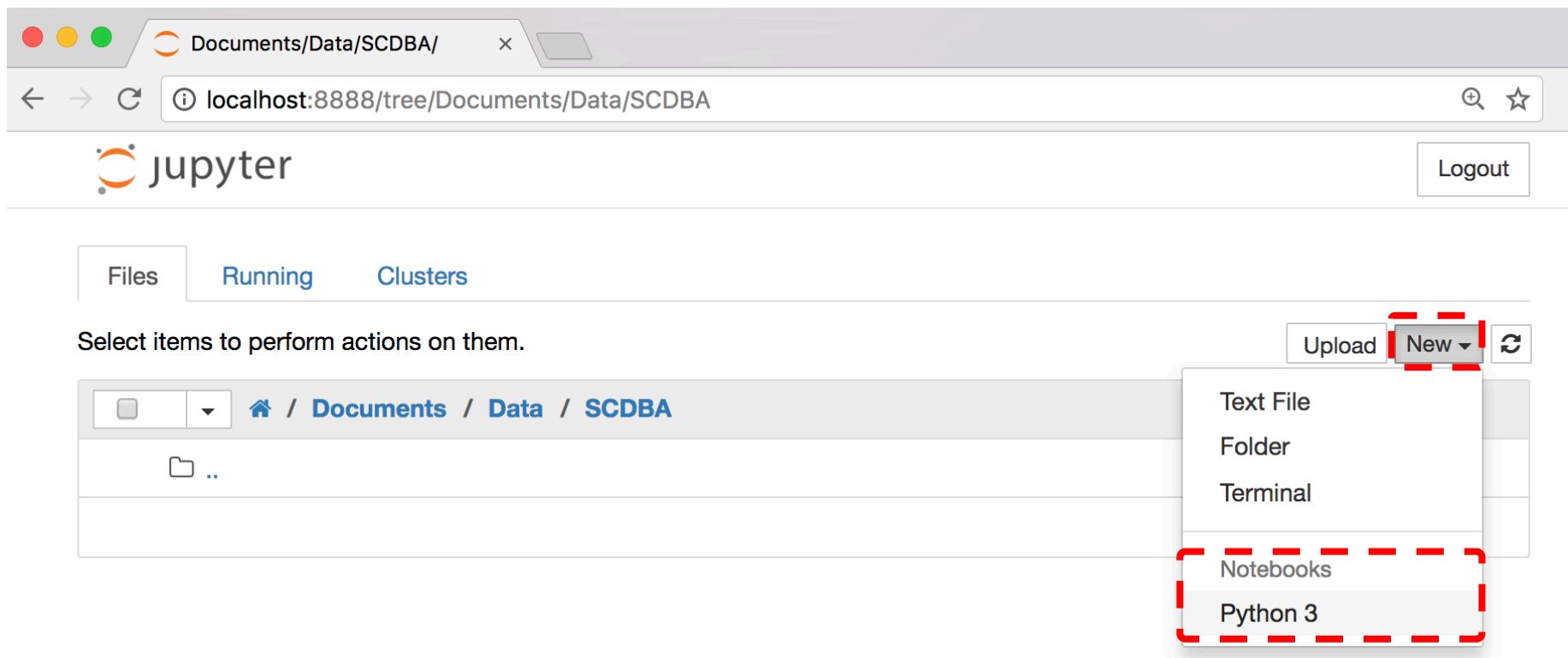
The screenshot shows the Anaconda Navigator application window. At the top, there's a navigation bar with the Anaconda Navigator logo, a search bar labeled "Applications on root", and buttons for "Upgrade Now" and "Sign in to Anaconda Cloud". On the left, a sidebar menu includes "Home", "Environments", "Projects (beta)", "Learning", "Community", "Documentation", "Developer Blog", and "Feedback". Below the sidebar, social media links for Twitter, YouTube, and GitHub are present. The main content area displays a grid of application icons. A central modal dialog box is open, displaying a welcome message: "Thanks for installing Anaconda! Anaconda Navigator helps you easily start important Python applications and manage the packages in your local Anaconda installation. It also connects you to online resources for learning and engaging with the Python, SciPy, and PyData community. To help us improve Anaconda Navigator, fix bugs, and make it even easier for everyone to use Python, we gather anonymized usage information, just like most web browsers and mobile apps. To opt out of this, please uncheck below (You can always change this setting in the Preferences menu)." There is a checked checkbox for "Yes, I'd like to help improve Anaconda." and two buttons at the bottom: "Ok" and "Ok, and don't show again". The "Ok, and don't show again" button has a red dashed border around it. The application icons include Jupyter Notebook (4.3.1), spyder (3.1.2), anaconda-fusion (1.0.2), glueviz (0.9.1), and rstudio (1.0.136). Each icon has a "Launch" or "Install" button below it.

Jupyter Notebook

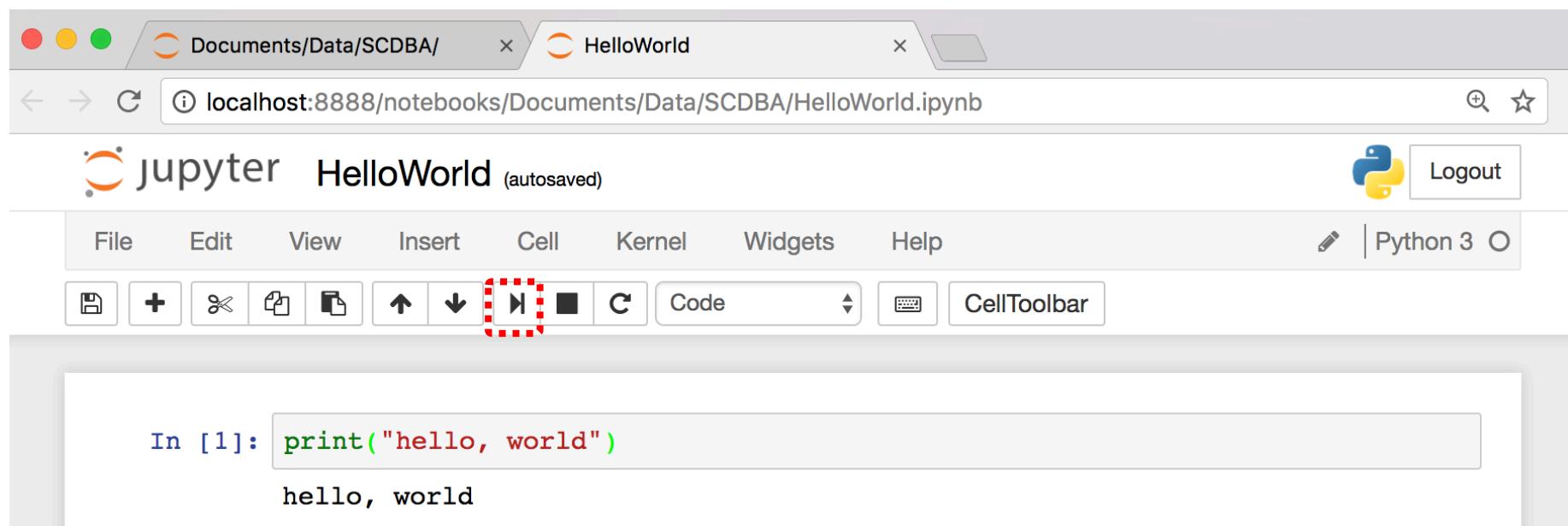
 <p>jupyter notebook  4.3.1</p> <p>Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.</p> <p>Launch</p>	 <p>qtconsole 4.2.1</p> <p>PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.</p> <p>Launch</p>	 <p>spyder  3.1.2</p> <p>Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features</p> <p>Launch</p>
 <p>anaconda-fusion 1.0.2</p> <p>Integration between Excel ® and Anaconda via Notebooks. Run data science functions, interact with results and create advanced visualizations in a code-free app inside Excel</p> <p>Install</p>	 <p>glueviz 0.9.1</p> <p>Multidimensional data visualization across files. Explore relationships within and among related datasets.</p> <p>Install</p>	 <p>rstudio 1.0.136</p> <p>A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.</p> <p>Install</p>

Jupyter Notebook

New Python 3



print("hello, world")



```
from platform import python_version
print("Python Version:", python_version())
```

The screenshot shows a Jupyter Notebook interface running in a web browser. The title bar indicates the notebook is titled "HelloWorld". The toolbar includes standard file operations like Open, Save, and New, along with a Cell toolbar specifically for the current cell.

In [1]: `print("hello, world")`
hello, world

In [2]: `from platform import python_version
print("Python Version:", python_version())`
Python Version: 3.6.0

Create Python Environments with Anaconda

- Python 3.6
- Python 3.5
 - Python 3.5.3
 - Python 3.5.2
- Python 2.7

Anaconda Create New Python 3.5 Environment (py35)

ANACONDA NAVIGATOR

Sign in to Anaconda Cloud

Home Environments Projects (beta) Learning Community Documentation Developer Blog Feedback

Search Environments root

Installed Channels Update index... Search Packages

Create new environment

Environment name: py35

Python version: 3.5

Create Cancel

186 packages available (root)

py35 Python 3.5

The screenshot shows the Anaconda Navigator interface. On the left, there's a sidebar with links for Home, Environments (which is selected and highlighted with a red dashed border), Projects (beta), Learning, Community, Documentation, Developer Blog, and Feedback. Below the sidebar are social media icons for Twitter, YouTube, and GitHub. The main area has tabs for Installed, Channels, and Update index... At the top right is a 'Sign in to Anaconda Cloud' button. In the center, there's a search bar for environments and a table of packages. A modal window titled 'Create new environment' is open, showing fields for 'Environment name' (set to 'py35'), 'Python' (checked), 'R' (unchecked), and 'Python version' (set to '3.5'). Below the modal, the package list starts with '_license' (version 1.1). Other packages listed include astroid, astropy, babel, backports, backports.shutil-get-terminal-size, and beautifulsoup4. At the bottom of the package list, it says '186 packages available (root)'. A large red text overlay 'py35 Python 3.5' is overlaid on the left side of the central area.

Anaconda Create New Python 2.7 Environment (py27)

The screenshot shows the Anaconda Navigator interface. On the left, there's a sidebar with icons for Home, Environments (highlighted with a red dashed box), Projects (beta), Learning, Community, Documentation, Developer Blog, and Feedback. Below the sidebar are social media links for Twitter, YouTube, and GitHub.

In the main area, there's a search bar for environments and a list of environments: root and py35. A large red box highlights the 'py35' environment. To its right, a table lists installed packages: openssl, pip, python, readline, setuptools, sqlite, tk, wheel, xz, and zlib. The 'python' package is highlighted with a blue arrow icon.

On the far right, a modal window titled 'Create new environment' is open. It has fields for 'Environment name' (set to 'py27'), 'Python' (checked), 'R' (unchecked), and 'Python version' (set to '2.7'). There are 'Cancel' and 'Create' buttons at the bottom.

At the bottom of the main pane, it says '10 packages available (/Users/imyday/anaconda/envs/py35)'.

Red annotations are present: 'py35 Python 3.5' is written over the 'py35' environment entry, and 'py27 Python 2.7' is written over the 'Create new environment' modal.

Verify that conda is installed, check current conda version

- **conda --version**
- Update conda to the current version
 - **conda update conda**

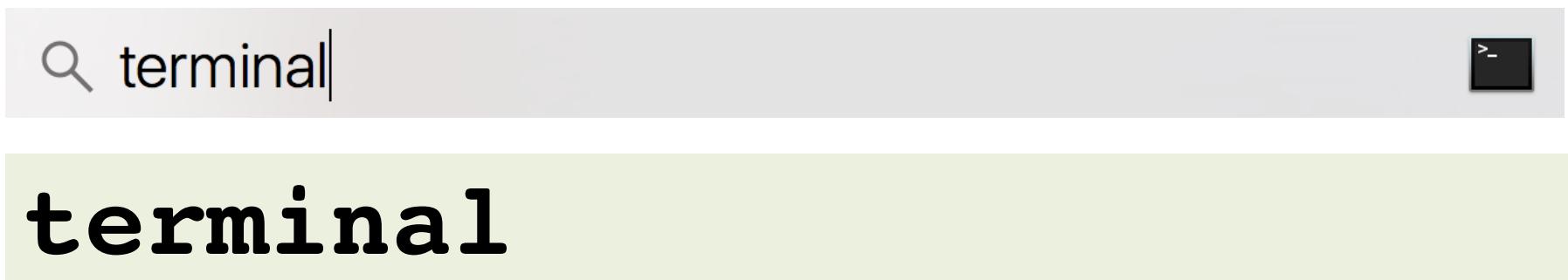
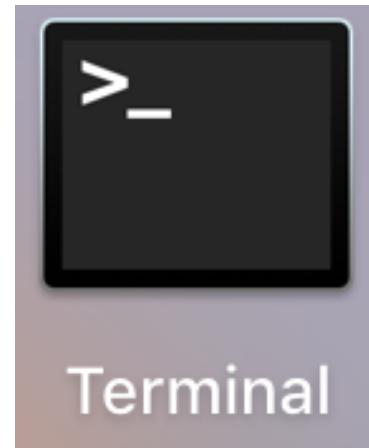
Check current conda version

Check current python version

Check conda environments

- **conda --version**
- **python --version**
- **conda info --envs**

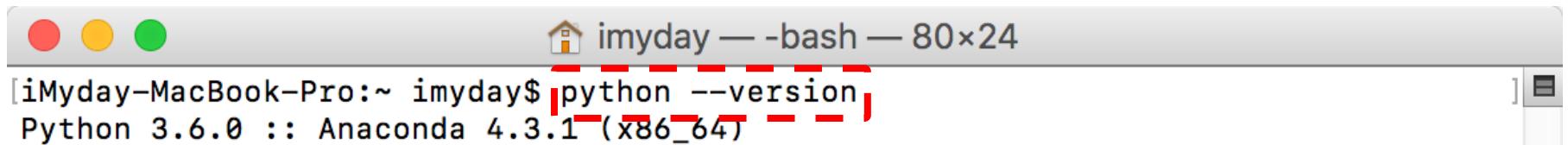
Terminal



conda list

```
iMyday-MacBook-Pro:~ imyday$ conda list
# packages in environment at /Users/imyday/anaconda:
#
_license           1.1                  py36_1
alabaster          0.7.9                py36_0
anaconda           4.3.1      np111py36_0
anaconda-client    1.6.0                py36_0
anaconda-navigator 1.5.0                py36_0
anaconda-project   0.4.1                py36_0
appnope             0.1.0                py36_0
appscript            1.0.1                py36_0
astroid              1.4.9                py36_0
astropy             1.3      np111py36_0
babel               2.3.4                py36_0
backports           1.0                  py36_0
beautifulsoup4     4.5.3                py36_0
bitarray             0.8.1                py36_0
blaze                0.10.1               py36_0
bokeh                0.12.4               py36_0
boto                 2.45.0                py36_0
bottleneck          1.2.0      np111py36_0
cffi                 1.9.1                py36_0
chardet              2.3.0                py36_0
chest                 0.2.3                py36_0
```

python --version



A screenshot of a macOS terminal window titled "imyday — -bash — 80x24". The window has red, yellow, and green close buttons. The command "python --version" is typed at the prompt, and the output "Python 3.6.0 :: Anaconda 4.3.1 (x86_64)" is displayed. The word "python" in the command and the first part of the output are highlighted with a red dashed underline.

```
[iMyday-MacBook-Pro:~ imyday$ python --version
Python 3.6.0 :: Anaconda 4.3.1 (x86_64)
```

conda --version

```
iMyday-MacBook-Pro:~ imyday$ python --version  
Python 3.6.0 :: Anaconda 4.3.1 (x86_64)  
[iMyday-MacBook-Pro:~ imyday$ conda --version  
conda 4.3.14  
[iMyday-MacBook-Pro:~ imyday$ conda info --envs  
# conda environments:  
#  
py27          /Users/imyday/anaconda/envs/py27  
py35          /Users/imyday/anaconda/envs/py35  
root          * /Users/imyday/anaconda
```

python --version
conda --version
conda info --envs

```
[iMyday-MacBook-Pro:~ imyday$ source activate py35  
[(py35) iMyday-MacBook-Pro:~ imyday$ python --version  
Python 3.5.3 :: Continuum Analytics, Inc.
```

source activate py35

```
[(py35) iMyday-MacBook-Pro:~ imyday$ conda --version  
conda 4.3.14
```

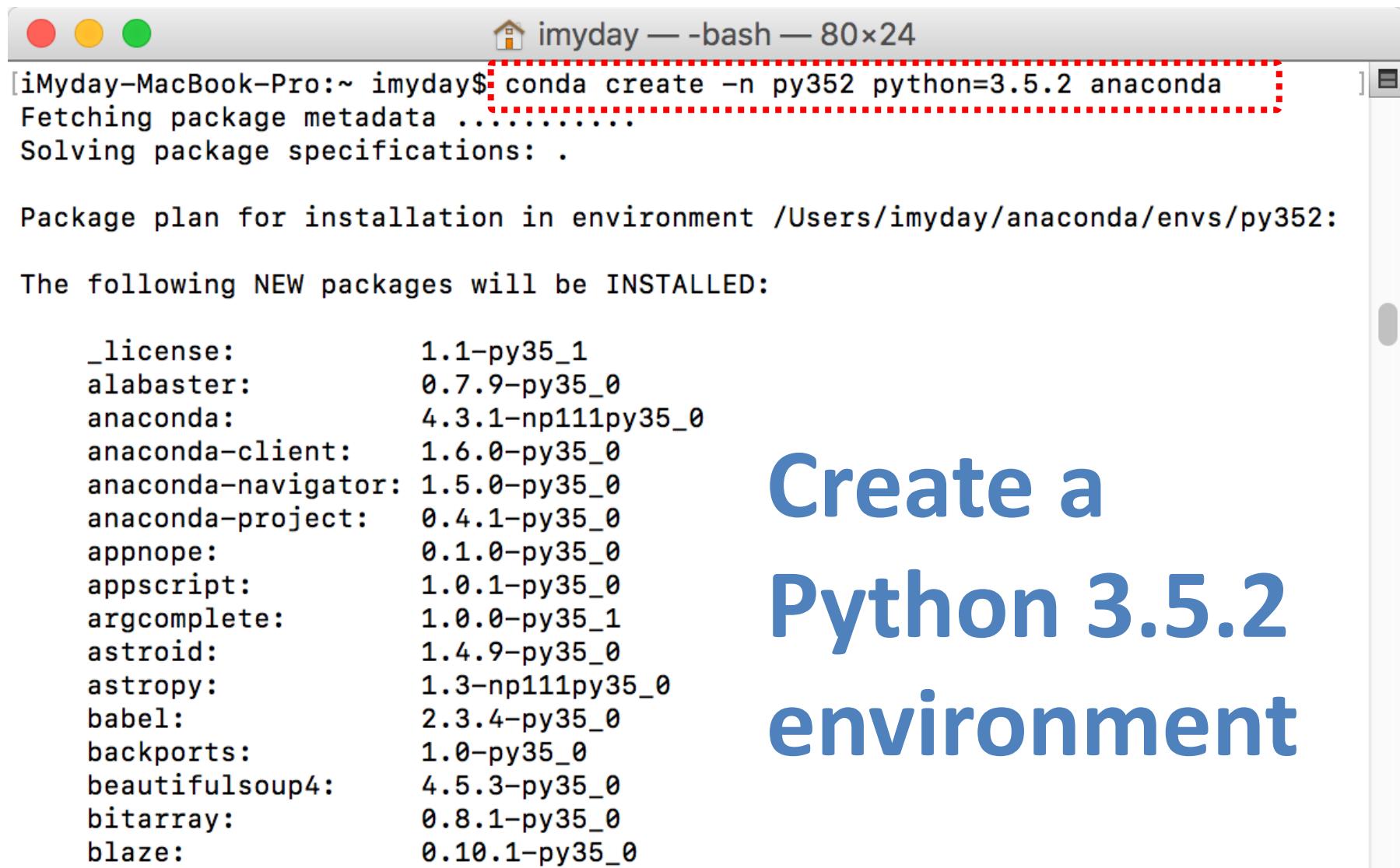
```
[(py35) iMyday-MacBook-Pro:~ imyday$ source deactivate py35
```

```
[iMyday-MacBook-Pro:~ imyday$ conda info --envs  
# conda environments:  
#
```

```
py27          /Users/imyday/anaconda/envs/py27  
py35          /Users/imyday/anaconda/envs/py35  
root          * /Users/imyday/anaconda
```

source deactivate py35

```
conda create -n py352 python=3.5.2 anaconda
```



A screenshot of a macOS terminal window titled "imyday — -bash — 80x24". The window shows the command "conda create -n py352 python=3.5.2 anaconda" being run. The output includes package metadata fetching, solving specifications, and a package plan for installation in the environment "/Users/imyday/anaconda/envs/py352". The terminal also lists the new packages that will be installed, including _license, alabaster, anaconda, anaconda-client, anaconda-navigator, anaconda-project, appnope, appscript, argcomplete, astroid, astropy, babel, backports, beautifulsoup4, bitarray, and blaze.

```
[iMyday-MacBook-Pro:~ imyday$ conda create -n py352 python=3.5.2 anaconda
Fetching package metadata .....
Solving package specifications: .

Package plan for installation in environment /Users/imyday/anaconda/envs/py352:

The following NEW packages will be INSTALLED:

_license:          1.1-py35_1
alabaster:         0.7.9-py35_0
anaconda:          4.3.1-np111py35_0
anaconda-client:   1.6.0-py35_0
anaconda-navigator: 1.5.0-py35_0
anaconda-project:  0.4.1-py35_0
appnope:           0.1.0-py35_0
appscript:          1.0.1-py35_0
argcomplete:        1.0.0-py35_1
astroid:            1.4.9-py35_0
astropy:            1.3-np111py35_0
babel:              2.3.4-py35_0
backports:          1.0-py35_0
beautifulsoup4:     4.5.3-py35_0
bitarray:           0.8.1-py35_0
blaze:              0.10.1-py35_0
```

Create a Python 3.5.2 environment

```
conda create -n py352 python=3.5.2 anaconda
```

```
i myday — -bash — 80x24
pyopenssl-16.2 100% | #####| Time: 0:00:00 1.40 MB/s
scikit-image-0 100% | #####| Time: 0:00:17 1.05 MB/s
seaborn-0.7.1- 100% | #####| Time: 0:00:00 1.05 MB/s
statsmodels-0. 100% | #####| Time: 0:00:04 1.06 MB/s
anaconda-navig 100% | #####| Time: 0:00:04 1.05 MB/s
blaze-0.10.1-p 100% | #####| Time: 0:00:00 1.05 MB/s
ipykernel-4.5. 100% | #####| Time: 0:00:00 1.21 MB/s
nbconvert-4.2. 100% | #####| Time: 0:00:00 1.22 MB/s
jupyter_consol 100% | #####| Time: 0:00:00 2.74 MB/s
notebook-4.3.1 100% | #####| Time: 0:00:05 1.05 MB/s
qtconsole-4.2. 100% | #####| Time: 0:00:00 1.03 MB/s
spyder-3.1.2-p 100% | #####| Time: 0:00:03 1.06 MB/s
widgetsnbexten 100% | #####| Time: 0:00:01 1.05 MB/s
ipywidgets-5.2 100% | #####| Time: 0:00:00 1.08 MB/s
jupyter-1.0.0- 100% | #####| Time: 0:00:00 2.53 MB/s
anaconda-4.3.1 100% | #####| Time: 0:00:00 4.49 MB/s
#
# To activate this environment, use:
# > source activate py352
#
# To deactivate this environment, use:
# > source deactivate py352
#
```

```
source activate py352
```

conda info --envs

```
[iMyday-MacBook-Pro:~ imyday$ conda info --envs
# conda environments:
#
py27          /Users/imyday/anaconda/envs/py27
py35          /Users/imyday/anaconda/envs/py35
py352         /Users/imyday/anaconda/envs/py352
root          * /Users/imyday/anaconda

[iMyday-MacBook-Pro:~ imyday$ python --version
Python 3.6.0 :: Anaconda 4.3.1 (x86_64)
[iMyday-MacBook-Pro:~ imyday$ source activate py352
(py352) iMyday-MacBook-Pro:~ imyday$ conda info --envs
# conda environments:
#
py27          /Users/imyday/anaconda/envs/py27
py35          /Users/imyday/anaconda/envs/py35
py352         * /Users/imyday/anaconda/envs/py352
root          /Users/imyday/anaconda

(py352) iMyday-MacBook-Pro:~ imyday$ python --version
Python 3.5.2 :: Anaconda 4.3.1 (x86_64)
(py352) iMyday-MacBook-Pro:~ imyday$ ]
```

```
conda info --envs
```

```
source activate py27
```

```
python --version
```

```
conda install notebook ipykernel
```

```
jupyter notebook
```

source activate py27

conda install notebook ipykernel

```
iMyday-MacBook-Pro:~ imyday$ conda info --envs
# conda environments:
#
py27          /Users/imyday/anaconda/envs/py27
py35          /Users/imyday/anaconda/envs/py35
py352         /Users/imyday/anaconda/envs/py352
root          * /Users/imyday/anaconda

[iMyday-MacBook-Pro:~ imyday$ source activate py27
[(py27) iMyday-MacBook-Pro:~ imyday$ python --version
Python 2.7.13 :: Continuum Analytics, Inc.
[(py27) iMyday-MacBook-Pro:~ imyday$ conda install notebook ipykernel
Fetching package metadata .....
Solving package specifications: .

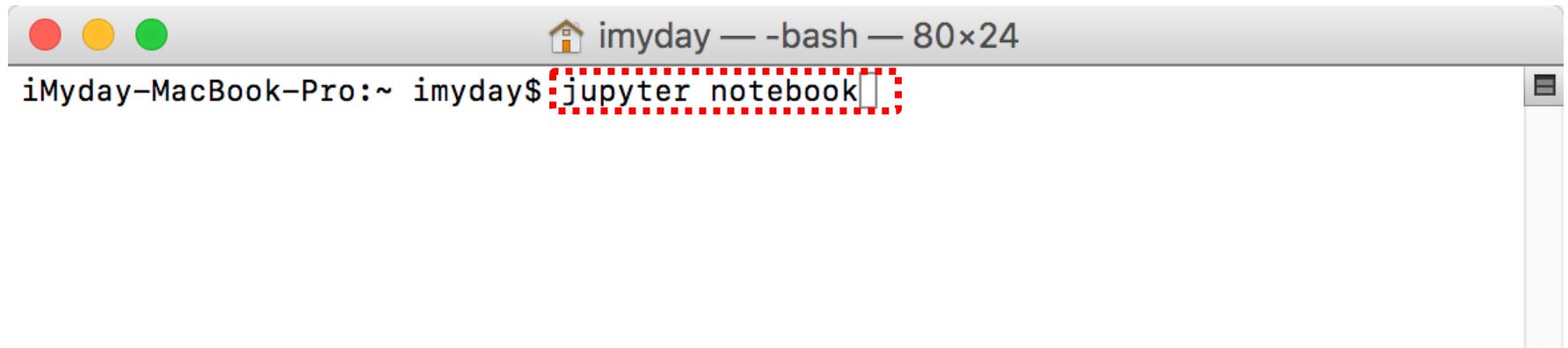
Package plan for installation in environment /Users/imyday/anaconda/envs/py27:

The following NEW packages will be INSTALLED:

appnope:      0.1.0-py27_0
backports:    1.0-py27_0
backports_abc: 0.5-py27_0
bleach:       1.5.0-py27_0
configparser: 3.5.0-py27_0
```

`conda info --envs
source activate py27
python --version
conda install notebook ipykernel
jupyter notebook`

jupyter notebook



A screenshot of a macOS terminal window. The title bar shows the path "imyday — -bash — 80x24". The command "jupyter notebook" is being typed into the terminal, with the last part "notebook" highlighted by a red dashed rectangle. The window has the standard OS X look with red, yellow, and green close buttons.

jupyter notebook
ipython notebook

jupyter notebook

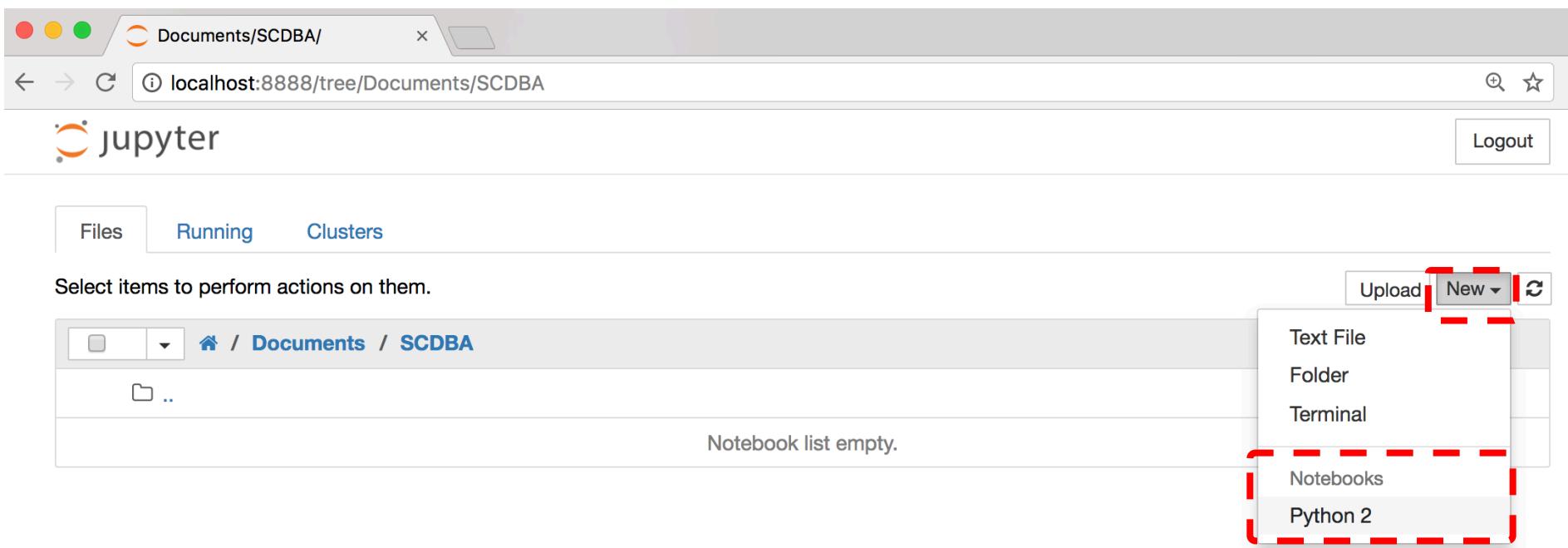
```
imyday — jupyter-notebook ▶ python — 80x24

[(py27) iMyday-MacBook-Pro:~ imyday$ jupyter notebook
[W 07:27:29.771 NotebookApp] Widgets are unavailable. Please install widgetsnbextension or ipywidgets 4.0
[I 07:27:29.808 NotebookApp] Serving notebooks from local directory: /Users/imyday
[I 07:27:29.808 NotebookApp] 0 active kernels
[I 07:27:29.808 NotebookApp] The Jupyter Notebook is running at: http://localhost:8888/?token=9cab2ca4b397ce9c4d48a4ef063ff235ffc7a1fc3a9d3ed6
[I 07:27:29.808 NotebookApp] Use Control-C to stop this server and shut down all
kernels (twice to skip confirmation).
[C 07:27:29.810 NotebookApp]

Copy/paste this URL into your browser when you connect for the first time,
to login with a token:
http://localhost:8888/?token=9cab2ca4b397ce9c4d48a4ef063ff235ffc7a1fc3a9d3ed6
[I 07:27:30.162 NotebookApp] Accepting one-time-token-authenticated connection from ::1
[W 07:42:52.367 NotebookApp] 404 GET /nbextensions/widgets/notebook/js/extension.js?v=20170315072729 (::1) 13.01ms referer=http://localhost:8888/notebooks/Documents/Data/SCDBA/HelloWorld.ipynb
[I 07:42:52.543 NotebookApp] Kernel started: 390583e8-e01f-448f-8e26-ecd96a630728
```

Jupyter Notebook

New Python 2



print "hello, world"

A screenshot of a Jupyter Notebook interface. At the top, there are three tabs: 'Documents/Data/SCDBA/' and 'HelloPython2'. Below the tabs is a URL bar showing 'localhost:8888/notebooks/Documents/Data/SCDBA/HelloPython2.ipynb'. The main title bar says 'jupyter HelloPython2 (unsaved changes)'. On the right side, there is a Python logo icon and a 'Logout' button. The menu bar includes 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', and 'Help'. The toolbar below the menu has icons for file operations like save, new, cut, copy, paste, and cell controls. A dropdown menu for 'Code' is open. The code cell content is 'In [1]: print "hello, world"', which has been executed and shows the output 'hello, world'.

```
In [1]: print "hello, world"
hello, world
```

```
from platform import python_version
print "Python Version:", python_version()
```

jupyter HelloPython2 (unsaved changes)



Logout

File Edit View Insert Cell Kernel Help

Python 2



```
In [1]: print "hello, world"
```

hello, world

```
In [2]: from platform import python_version
print "Python Version:", python_version()
```

Python Version: 2.7.13

jupyter notebook

ipython notebook



Text input and output

```
print("Hello World")
```

```
print("Hello World\nThis is a message")
```

```
x = 3  
print(x)
```

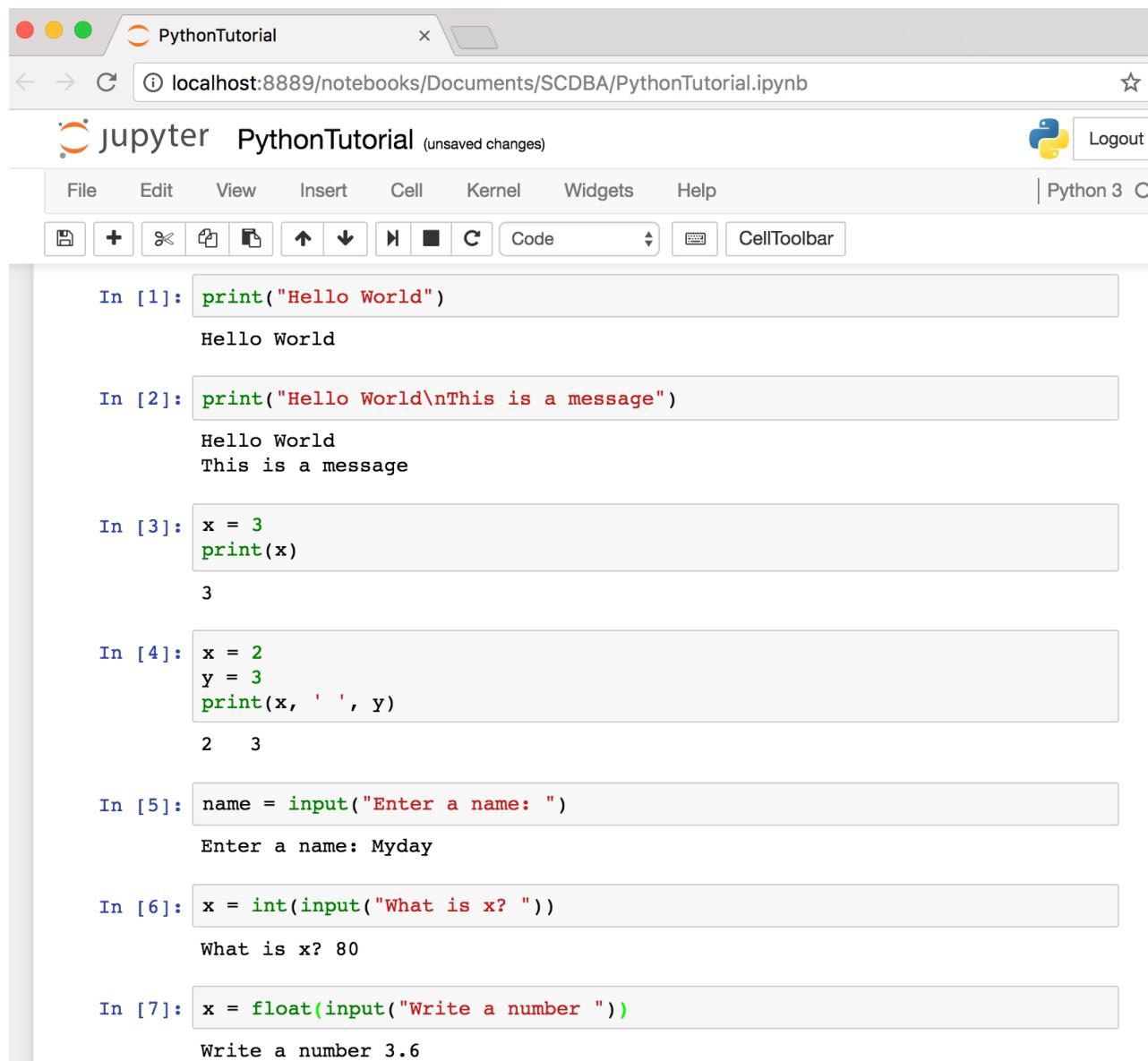
```
x = 2  
y = 3  
print(x, ' ', y)
```

```
name = input("Enter a name: ")
```

```
x = int(input("What is x? "))
```

```
x = float(input("Write a number "))
```

Text input and output



The screenshot shows a Jupyter Notebook interface titled "PythonTutorial" running on "localhost:8889/notebooks/Documents/SCDBA/PythonTutorial.ipynb". The notebook contains the following code cells:

- In [1]: `print("Hello World")`
Hello World
- In [2]: `print("Hello World\nThis is a message")`
Hello World
This is a message
- In [3]: `x = 3
print(x)`
3
- In [4]: `x = 2
y = 3
print(x, ' ', y)`
2 3
- In [5]: `name = input("Enter a name: ")`
Enter a name: Myday
- In [6]: `x = int(input("What is x? "))`
What is x? 80
- In [7]: `x = float(input("Write a number "))`
Write a number 3.6

Variables

```
x = 2  
price = 2.5  
word = 'Hello'
```

```
word = 'Hello'  
word = "Hello"  
word = '''Hello'''
```

```
x = 2  
x = x + 1  
x = 5
```

Python Basic Operators

```
print('7 + 2 =', 7 + 2)
print('7 - 2 =', 7 - 2)
print('7 * 2 =', 7 * 2)
print('7 / 2 =', 7 / 2)
print('7 // 2 =', 7 // 2)
print('7 % 2 =', 7 % 2)
print('7 ** 2 =', 7 ** 2)
```

```
print('7 + 2 =', 7 + 2)
print('7 - 2 =', 7 - 2)
print('7 * 2 =', 7 * 2)
print('7 / 2 =', 7 / 2)
print('7 // 2 =', 7 // 2)
print('7 % 2 =', 7 % 2)
print('7 ** 2 =', 7 ** 2)
```

7 + 2 = 9
7 - 2 = 5
7 * 2 = 14
7 / 2 = 3.5
7 // 2 = 3
7 % 2 = 1
7 ** 2 = 49

BMI Calculator in Python

```
height_cm = float(input("Enter your height in cm: "))
weight_kg = float(input("Enter your weight in kg: "))

height_m = height_cm/100
BMI = (weight_kg/(height_m**2))

print("Your BMI is: " + str(round(BMI,1)))
```

BMI Calculator in Python

jupyter PythonTutorial Last Checkpoint: a minute ago (unsaved changes)



File Edit View Insert Cell Kernel Widgets Help

Python 3



```
In [1]: height_cm = float(input("Enter your height in cm: "))
weight_kg = float(input("Enter your weight in kg: "))

height_m = height_cm/100
BMI = (weight_kg/(height_m**2))

print("Your BMI is: " + str(round(BMI,1)))
```

```
Enter your height in cm: 170
Enter your weight in kg: 60
Your BMI is: 20.8
```

```
In [ ]:
```

If statements

> greater than
< smaller than
== equals
!= is not

```
score = 80
if score >=60 :
    print("Pass")
else:
    print("Fail")
```

Pass

```
score = 80
if score >=60 :
    print("Pass")
else:
    print("Fail")
```

For loops

```
for i in range(1,11):  
    print(i)
```

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```

For loops

```
for i in range(1,10):
    for j in range(1,10):
        print(i, ' * ', j, ' = ', i*j)
```

```
9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
9 * 4 = 36
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81
```

Functions

```
def convertCMTOM( xcm ) :  
    m = xcm/100  
    return m  
  
cm = 180  
m = convertCMTOM( cm )  
print( str(m) )
```

Lists

```
x = [60, 70, 80, 90]
print(len(x))
print(x[0])
print(x[1])
print(x[-1])
```

4
60
70
90

Tuples

A **tuple** in Python is a collection that cannot be modified.

A tuple is defined using **parenthesis**.

```
x = (10, 20, 30, 40, 50)
```

```
print(x[0])
```

10

```
print(x[1])
```

20

```
print(x[2])
```

30

```
print(x[-1])
```

50

Dictionary

```
k = { 'EN': 'English', 'FR': 'French' }  
print(k['EN'])
```

Dictionary

'EN' → 'English'

'FR' → 'French'

English

Sets

```
animals = {'cat', 'dog'}
```

```
animals = {'cat', 'dog'}
print('cat' in animals)    # Check if an element is in a set; prints "True"
print('fish' in animals)   # prints "False"
animals.add('fish')        # Add an element to a set
print('fish' in animals)   # Prints "True"
print(len(animals))        # Number of elements in a set; prints "3"
animals.add('cat')         # Adding an element that is already in the set does nothing
print(len(animals))        # Prints "3"
animals.remove('cat')      # Remove an element from a set
print(len(animals))        # Prints "2"
```

```
True
False
True
3
3
2
```

```
animals = {'cat', 'dog'}
print('cat' in animals)
print('fish' in animals)
animals.add('fish')
print('fish' in animals)
print(len(animals))
animals.add('cat')
print(len(animals))
animals.remove('cat')
print(len(animals))
```

Python Ecosystem

Python Ecosystem

import math

```
x = log(1)  
print(x)
```

```
-----  
NameError                                 Traceback (most recent call last)  
<ipython-input-64-55d85b4998db> in <module>()  
----> 1 x = log(1)  
      2 print(x)  
  
NameError: name 'log' is not defined
```

math.log?

```
import math  
x = math.log(1)  
print(x)
```

0.0

```
math.log(8,2)|
```

3.0

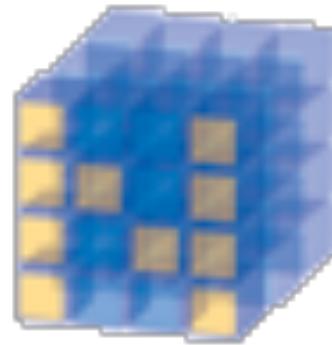
```
Docstring:  
log(x[, base])
```

Return the logarithm of x to the given base.

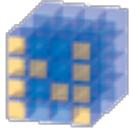
If the base not specified, returns the natural logarithm (base e) of x.

Type: builtin_function_or_method

Numpy



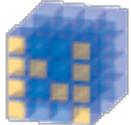
NumPy
Base
N-dimensional array
package



NumPy

NumPy

- NumPy provides a multidimensional array object to store homogenous or heterogeneous data; it also provides optimized functions/methods to operate on this array object.



NumPy

NumPy

```
v = range(1, 6)
```

```
print(v)
```

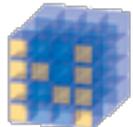
```
2 * v
```

```
import numpy as np
```

```
v = np.arange(1, 6)
```

```
v
```

```
2 * v
```



NumPy

Base

N-dimensional
array package

```
v = range(1, 6)
print(v)
```

```
[1, 2, 3, 4, 5]
```

```
2 * v
```

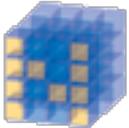
```
[1, 2, 3, 4, 5, 1, 2, 3, 4, 5]
```

```
import numpy as np
v = np.arange(1, 6)
v
```

```
array([1, 2, 3, 4, 5])
```

```
2 * v
```

```
array([ 2,  4,  6,  8, 10])
```



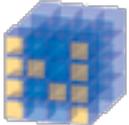
NumPy

NumPy

```
import numpy as np  
a = np.array([1, 2, 3])  
b = np.array([4, 5, 6])  
c = a * b  
c
```

```
import numpy as np  
a = np.array([1, 2, 3])  
b = np.array([4, 5, 6])  
c = a * b  
c
```

```
array([ 4, 10, 18])
```



NumPy

NumPy

```
import numpy as np

a = np.zeros((2,2))      # Create an array of all zeros
print(a)                  # Prints "[[ 0.  0.]
                           #           [ 0.  0.]]"

b = np.ones((1,2))       # Create an array of all ones
print(b)                  # Prints "[[ 1.  1.]]"

c = np.full((2,2), 7)    # Create a constant array
print(c)                  # Prints "[[ 7.  7.]
                           #           [ 7.  7.]]"

d = np.eye(2)             # Create a 2x2 identity matrix
print(d)                  # Prints "[[ 1.  0.]
                           #           [ 0.  1.]]"

e = np.random.random((2,2))# Create an array filled with random values
print(e)                  # Might print "[[ 0.91940167  0.08143941]
                           #           [ 0.68744134  0.87236687]]"
```

```
[[ 0.  0.]
 [ 0.  0.]]
[[ 1.  1.]]
[[ 7.  7.]
 [ 7.  7.]]
[[ 1.  0.]
 [ 0.  1.]]
[[ 0.22886991  0.68473232]
 [ 0.20683825  0.16589995]]
```

Compatible Python 2 and Python 3 Code

- `print()`
- Exceptions
- Division
- Unicode strings
- Bad imports

Compatible Python 2 and Python 3 Code

```
print()  
print("This works in py2 and py3")  
  
from __future__ import print_function  
print("Hello", "World")
```

Create Python 2 or 3 environments

The screenshot shows a web browser displaying the Conda documentation at conda.pydata.org/docs/py2or3.html#create-python-2-or-3-environments. The page title is "Create Python 2 or 3 environments". The left sidebar has a dark background and lists various Conda-related topics: "Conda", "Search docs", "Get started", "Using conda" (selected), "Managing environments", "Managing Python" (selected), "Managing packages", "Using R with conda", "Using Microsoft R Open", "Building packages", "Help & reference", and "Get involved". The main content area has a light background and discusses creating Python environments. It includes sections for "Create a Python 3.5 environment" and "Create a Python 2.7 environment", each with a code example in a terminal window.

Conda supports Python 2.7, 3.4, and 3.5. The default is Python 2.7 or 3.5, depending on which installer you used. If the installer you used is Anaconda or Miniconda, the default is 2.7. If the installer you used is Anaconda3 or Miniconda3, the default is 3.5.

Create a Python 3.5 environment

To create a new environment with a different version of Python, use the `conda create` command. In this example, we'll make the new environment for Python 3.5:

```
$ conda create -n py35 python=3.5 anaconda
```

Here, the 'py35' is the name of the environment you want to create, and 'anaconda' is the meta-package that includes all of the actual Python packages comprising the Anaconda distribution. When creating a new environment and installing Anaconda, you can specify the exact package and Python versions, for example, `numpy=1.7` or `python=3.5`.

Create a Python 2.7 environment

In this example, we'll make a new environment for Python 2.7:

```
$ conda create -n py27 python=2.7 anaconda
```

File IO with open()

```
# Python 2 only
f = open('myfile.txt')
data = f.read()                      # as a byte string
text = data.decode('utf-8')

# Python 2 and 3: alternative 1
from io import open
f = open('myfile.txt', 'rb')
data = f.read()                      # as bytes
text = data.decode('utf-8')          # unicode, not bytes

# Python 2 and 3: alternative 2
from io import open
f = open('myfile.txt', encoding='utf-8')
text = f.read()          # unicode, not bytes
```

Anaconda Cloud

CONTINUUM[®]
ANALYTICS

Gallery

About

Pricing

Anaconda

Help

Download Anaconda

Sign In

Search Anaconda Cloud



Where packages, notebooks, and environments are shared.

Powerful collaboration and package management for open source and private projects.

Public projects and notebooks are always free.

Private plans start at \$7/month.

Sign Up

Sign In

New to Anaconda Cloud? Sign up!

Pick a username

Your email

Use at least one lowercase letter, one numeral, and seven characters.

Create a password

Confirm password

I accept the [Terms & Conditions](#)

Sign up!

By clicking "Sign up!" you agree to our privacy policy and terms of service. We will send you account related emails occasionally.

Why you'll love Anaconda Cloud

Making it easy to share packages, notebooks, and environments to be more collaborative.

<https://anaconda.org/>

Conda Get-Started

The screenshot shows the Conda documentation website. The sidebar on the left includes links for "Conda Announcements List", "Get started" (which is expanded to show "Intro to conda", "Download conda", "Installation", "Test drive", and "Conda cheat sheet"), "Using conda", "Building packages", "Help & reference", and "Get involved". The main content area shows the "Get started" page with a breadcrumb trail "Docs » Get started" and a "Edit on GitHub" button. The page title is "Get started" and contains a bulleted list of topics related to getting started with Conda.

- [Intro to conda](#)
- [Download conda](#)
 - [Should I download Anaconda or Miniconda?](#)
 - [Which version of Anaconda or Miniconda should I choose?](#)
 - [Should I choose GUI installer or command line installer?](#)
 - [What version of Python should I choose?](#)
 - [What about cryptographic hash verification?](#)
- [Installation](#)
 - [Quick install](#)
 - [Miniconda quick install requirements](#)
 - [Windows Miniconda install](#)
 - [OS X Miniconda install](#)
 - [Linux Miniconda install](#)
 - [Full installation](#)
 - [Anaconda requirements](#)
 - [Install instructions](#)
 - [Windows Anaconda install](#)
 - [OS X Anaconda install](#)
 - [Linux Anaconda install](#)
 - [Configuration](#)
 - [The conda configuration file \(.condarc\)](#)
 - [General configuration](#)

<http://conda.pydata.org/docs/get-started.html>

Python–Future

python-future.org/index.html

Python-Future Overview Cheat Sheet FAQ Contents ▾ Page ▾ Search

Easy, clean, reliable Python 2/3 compatibility

Table of Contents

What's New

Overview: Easy, clean, reliable Python 2/3 compatibility

Quick-start guide

Cheat Sheet: Writing Python 2/3 compatible code

Imports

What else you need to know

Automatic conversion to Py2/3

Frequently Asked Questions (FAQ)

Standard library incompatibilities

Older interfaces

Changes in previous versions

Licensing and credits

API Reference (in progress)

Easy, clean, reliable Python 2/3 compatibility

`python-future` is the missing compatibility layer between Python 2 and Python 3. It allows you to use a single, clean Python 3.x-compatible codebase to support both Python 2 and Python 3 with minimal overhead.

Contents:

- [What's New](#)
 - [What's new in version 0.15.2 \(2015-09-11\)](#)
 - [What's new in version 0.15.1 \(2015-09-09\)](#)
 - [What's new in version 0.15.0 \(2015-07-25\)](#)
 - [Previous versions](#)
- [Overview: Easy, clean, reliable Python 2/3 compatibility](#)
 - [Features](#)
 - [Code examples](#)
 - [Automatic conversion to Py2/3-compatible code](#)
 - [Automatic translation](#)
 - [Licensing](#)
 - [Next steps](#)
- [Quick-start guide](#)
 - [Installation](#)
 - [If you are writing code from scratch](#)
 - [To convert existing Python 3 code](#)
 - [To convert existing Python 2 code](#)
 - [Standard library reorganization](#)
 - [Python 2-only dependencies](#)
 - [Next steps](#)
- [Cheat Sheet: Writing Python 2/3 compatible code](#)
 - [Setup](#)
 - [Essential syntax differences](#)
 - [Strings and bytes](#)

<http://python-future.org/index.html>

Fork me on GitHub

pip install future

pip install six

The imports below refer to these pip-installable packages on PyPI:

```
import future          # pip install future
import builtins        # pip install future
import past            # pip install future
import six             # pip install six
```

```
futurize              # pip install future
pasteurize            # pip install future
```

print

```
# Python 2 only:  
print 'Hello'
```

```
# Python 2 and 3:  
print('Hello')
```

```
# Python 2 only:  
print 'Hello', 'Guido'
```

```
# Python 2 and 3:  
from __future__ import print_function #(at top of module)  
  
print('Hello', 'Guido')
```

Writing Python 2-3 compatible code

Essential syntax differences

print

```
# Python 2 only:  
print 'Hello'
```

```
# Python 2 and 3:  
print('Hello')
```

To print multiple strings, import `print_function` to prevent Py2 from interpreting it as a tuple:

```
# Python 2 only:  
print 'Hello', 'Guido'
```

```
# Python 2 and 3:  
from __future__ import print_function    # (at top of module)  
  
print('Hello', 'Guido')
```

Unicode (text) string literals

```
# Python 2 only
s1 = 'The Zen of Python'
s2 = u'きたないのよりきれいな方がいい\n'

# Python 2 and 3
s1 = u'The Zen of Python'
s2 = u'きたないのよりきれいな方がいい\n'
```

Unicode (text) string literals

```
# Python 2 and 3
from __future__ import unicode_literals # at top of module

s1 = 'The Zen of Python'
s2 = 'きたないのよりきれいな方がいい\n'
```

Six: Python 2 and 3 Compatibility Library

← → C https://pythonhosted.org/six/ ⌂ ⌂ ⌂

modules | index

six 1.10.0 documentation »

Table Of Contents

- Six: Python 2 and 3 Compatibility Library
 - Indices and tables
 - Package contents
 - Constants
 - Object model compatibility
 - Syntax compatibility
 - Binary and text data
 - unittest assertions
 - Renamed modules and attributes compatibility
 - urllib parse
 - urllib error
 - urllib request
 - urllib response
 - Advanced - Customizing renames

Six: Python 2 and 3 Compatibility Library

Six provides simple utilities for wrapping over differences between Python 2 and Python 3. It is intended to support codebases that work on both Python 2 and 3 without modification. six consists of only one Python file, so it is painless to copy into a project.

Six can be downloaded on PyPi. Its bug tracker and code hosting is on BitBucket.

The name, “six”, comes from the fact that 2×3 equals 6. Why not addition? Multiplication is more powerful, and, anyway, “five” has already been snatched away by the (admittedly now moribund) Zope Five project.

Indices and tables

- Index
- Search Page

Package contents

six.PY2
A boolean indicating if the code is running on Python 2.

six.PY3
A boolean indicating if the code is running on Python 3.

Constants

Six provides constants that may differ between Python versions. Ones ending `_types` are mostly useful as the second argument to `isinstance` OR `issubclass`.

six.class_types
Possible class types. In Python 2, this encompasses old-style and new-style classes. In Python 3, this is just new-styles.

Conda Test Drive

The screenshot shows a web browser displaying the Conda Test Drive documentation. The URL in the address bar is conda.pydata.org/docs/test-drive.html. The page has a green header bar with the Conda logo and a search bar. The main content area has a breadcrumb navigation path: Docs > Get started > Test drive. There is a link to "Edit on GitHub". The main title is "Test drive". Below it, a text block says: "To start the conda 30-minute test drive, you should have already followed our 2-minute *Quick install* guide to download, install and update Miniconda, OR have downloaded, installed and updated Anaconda or Miniconda on your own." A note below states: "NOTE: After installing, be sure you have closed and then re-opened the terminal window so the changes can take effect." A section titled "Conda test drive milestones:" lists five numbered steps: 1. USING CONDA, 2. MANAGING ENVIRONMENTS, 3. MANAGING PYTHON, 4. MANAGING PACKAGES, and 5. REMOVING PACKAGES, ENVIRONMENTS, OR CONDA. The sidebar on the left contains a navigation menu with links to "Get started", "Intro to conda", "Download conda", "Installation", "Test drive" (which is highlighted), "Conda cheat sheet", "Using conda", "Building packages", "Help & reference", and "Get involved".

Test drive

To start the conda 30-minute test drive, you should have already followed our 2-minute *Quick install* guide to download, install and update Miniconda, OR have downloaded, installed and updated Anaconda or Miniconda on your own.

NOTE: After installing, be sure you have closed and then re-opened the terminal window so the changes can take effect.

Conda test drive milestones:

1. **USING CONDA.** First we will verify that you have installed Anaconda or Miniconda, and check that it is updated to the current version. 3 min.
2. **MANAGING ENVIRONMENTS.** Next we will play with environments by creating a few environments, so you can learn to move easily between the environments. We will also verify which environment you are in, and make an exact copy of an environment as a backup. 10 min.
3. **MANAGING PYTHON.** Then we will check to see which versions of Python are available to install, install another version of Python, and switch between versions. 4 min.
4. **MANAGING PACKAGES.** We play with packages. We will a) list packages installed on your computer, b) see a list of available packages, and c) install and remove some packages using conda install. For packages not available using conda install, we will d) search on Anaconda.org. For packages that are in neither location, we'll e) install a package with the pip package manager. We will also install a free 30 day trial of Continuum's commercial package IOPro. 10 min.
5. **REMOVING PACKAGES, ENVIRONMENTS, OR CONDA.** We'll end the test drive by removing

<http://conda.pydata.org/docs/test-drive.html>

Managing Conda and Anaconda

Managing conda and anaconda

conda info

Verify conda is installed, check version #

conda update conda

Update conda package and environment manager to current version

conda update anaconda

Update the anaconda meta package (the library of packages ready to install with **conda** command)

Managing environments

Managing environments

conda info --envs or **conda info -e** Get a list of all my environments, active environment shown with *

conda create --name snowflakes biopython Create an environment and install program(s)

or

conda create -n snowflakes biopython

TIP: To avoid dependency conflicts, install all programs in the environment (snowflakes) at the same time.

TIP: Environments install by default into the `envs` directory in your `conda` directory. You can specify a different path; see **conda create --help** for details.

source activate snowflakes (Linux, Mac)

Activate the new environment to use it

activate snowflakes (Windows)

TIP: `Activate` prepends the path to the `snowflakes` environment.

conda create -n bunnies python=3.4 astroid Create a new environment, specify Python version

conda create -n flowers --clone snowflakes Make exact copy of an environment

conda remove -n flowers --all

Delete an environment

conda env export > puppies.yml

Save current environment to a file

conda env create -f puppies.yml

Load environment fromm a file

Managing Python

Managing Python

conda search --full-name python

or

conda search -f python

Check versions of Python available to install

conda create -n snakes python=3.4

Install different version of Python in new environment

source activate snakes (*Linux, Mac*)

activate snakes (*Windows*)

Switch to the new environment that has a
different version of Python

TIP: *Activate prepends the path to the snakes environment.*

Managing Packages in Python

Managing packages, including Python

`conda list`

View list of packages and versions installed in active environment

`conda search beautiful-soup`

Search for a package to see if it is available to conda install

`conda install -n bunnies beautiful-soup` Install a new package

NOTE: If you do not include the name of the new environment (`-n bunnies`) it will install in the current active environment.

TIP: To view list of all packages available through `conda install`, visit <http://docs.continuum.io/anaconda/pkg-docs.html>.

`conda update beautiful-soup`

Update a package in the current environment

`conda search --override-channels -c pandas bottleneck` Search for a package in a specific location (i.e. the pandas channel on Anaconda.org)

NOTE: Or go to Anaconda.org in the browser and search by package name. This will show the specific channel (owner) through which it is available.

`conda install -c pandas bottleneck` Install a package from a specific channel

`conda search --override-channels -c defaults beautiful-soup` Search for a package to see if it is available from the Anaconda repository

`source activate bunnies` (Linux, Mac)

`activate bunnies` (Windows)

`pip install see`

Activate the environment where you want to install a package and install it with pip (included with Anaconda and Miniconda)

`conda install iopro accelerate`

Install commercial Continuum packages

`conda skeleton pypi pyinstrument`

`conda build pyinstrument`

Build a Conda package from a Python Package Index (PyPI) Package

PyCharm: Python IDE

A screenshot of the PyCharm website homepage. At the top, there's a navigation bar with links for 'ALL TOOLS', 'IDEs', '.NET & VISUAL STUDIO', 'TEAM TOOLS', 'LANGUAGES', 'STORE', 'SUPPORT', and 'WE ARE JETBRAINS'. On the right side of the bar are icons for user profile (0 notifications), search, and a star. The main content area features the 'PyCharm' logo with a stylized 'PC' icon, followed by the text 'PyCharm' and 'Python IDE for Professional Developers'. Below this is a 'DOWNLOAD NOW' button and a note about full-fledged professional or free community editions. A large blue 'Download' button is located on the right.

<http://www.jetbrains.com/pycharm/>

Python Fiddle

Python Cloud IDE | Python Fiddle ×

← → ⌂ ⓘ pythonfiddle.com ⋮

Run Reset Share Import Login Language▼

G+1 2.6k

Python Fiddle Python Cloud IDE

Examples

- [Chaining comparison operators](#)
- [Decorators](#)
- [Creating generators objects](#)
- [Enumerate](#)
- [Function closure](#)
- [Lex tokenizer](#)
- [Step argument in slice operators](#)
- [For Else](#)
- [Verbose regular expressions](#)
- [In-place value swapping](#)
- [Function argument unpacking](#)

Packages

Hotkeys

```
1 print("Hello Python Fiddle")
2
```

Title:

Description:

Tags: A comma-separated list of tags.

Save

Hello Python Fiddle

Python 2 or 3

Create Python 2 or 3 environments

Anaconda supports Python 2.7, 3.4, and 3.5. The default is Python 2.7 or 3.5, depending on which installer you used. If the installer you used is Anaconda or Miniconda, the default is 2.7. If the installer you used is Anaconda3 or Miniconda3, the default is 3.5.

Create a Python 3.5 environment [%](#)

To create a new environment with a different version of Python, use the `conda create` command. In this example, we'll make the new environment for Python 3.5:

```
$ conda create -n py35 python=3.5 anaconda
```

Here, the 'py35' is the name of the environment you want to create, and 'anaconda' is the meta-package that includes all of the actual Python packages comprising the Anaconda distribution. When creating a new environment and installing Anaconda, you can specify the exact package and Python versions, for example, `numpy=1.7` or `python=3.5`.

Create a Python 2.7 environment

In this example, we'll make a new environment for Python 2.7:

```
$ conda create -n py27 python=2.7 anaconda
```

References

- Yves Hilpisch (2014), Python for Finance: Analyze Big Financial Data, O'Reilly
- Ivan Idris (2015), Numpy Beginner's Guide, Third Edition, Packt Publishing
- Python, <https://www.python.org/>
- Python Programming Language, <http://pythonprogramminglanguage.com/>
- Numpy, <http://www.numpy.org/>