python™

Introduction to Python:
What is Python and Why is Python?

Mehmet Tevfik DORAK, MD PhD
Kingston University London
U.K.

Faculty of
Science,
Engineering
and Computing

Kingston
University
London

Schedule

Day 2
Introduction to Python as a coding language for bioinformatics
What can you do with programming languages?
What is Python and why Python?
Python basics
Simple Python codes
Demonstrations of more advanced methods
Questions & answers

Aim

Most programming books are written by computer scientists almost for other computing scientists who almost exclusively use Linux

If you are a biologist and a Windows user, you could easily get lost at page 1 of those books

Here is my take on learning Python as a biologist and an absolute beginner in programming who has no computer science degree, and never used Linux

Outline

- To make you appreciate the power of computer programming
- To equip you with sufficient amount of basic knowledge on Python, so that:

  - You can use a cookbook

  - You can use existing packages

  - You can communicate effectively with a programmer

  - To enable you to advance your knowledge in Python with ease

TOOLBOX

PICK UP PYTHON

A powerful programming language with huge community support.

ILLUSTRATION BY THE PROJECT TWINS

BY JEFFREY M. PERKEL

Last month, Adina Howe took up a post at Iowa State University in Ames. Officially, she is an assistant professor of agricultural and biosystems engineering. But she works not in the greenhouse, but in front of a keyboard. Howe is a programmer, and a key part of her job is as a 'data professor' — developing curricula to teach the next generation of graduates about the mechanics and importance of scientific programming.

Howe does not have a degree in computer science, nor does she have years of formal training. She had a PhD in environmental engineering and expertise in running enzyme assays when she joined the laboratory of Titus Brown at Michigan State University in East Lansing. Brown specializes in bioinformatics and uses computation to extract meaning from genomic data sets, and Howe had to get up to speed on the computational side. Brown's recommendation: learn Python.

Among the host of computer-programming languages that scientists might choose to pick up, Python, first released in 1991 by Dutch programmer Guido van Rossum, is an increasingly popular (and free) recommendation. It combines simple syntax, abundant online resources and a rich ecosystem of scientifically focused toolkits with a heavy emphasis on community.

HELLO, WORLD

With the explosive growth of 'big data' in disciplines such as bioinformatics, neuroscience and astronomy, programming know-how is becoming ever more crucial. Researchers who can write code in Python can deftly manage their data sets, and work much more efficiently on a whole host of research-related tasks — from crunching numbers to cleaning up, analysing and visualizing data. Whereas some programming languages, such as MATLAB and R, focus on mathematical and statistical operations, Python is a general-purpose language, along the lines of C and C++ (the languages in which much commercial software and operating systems are written). As such, it is perhaps more complicated, Brown says, but also more capable: it is amenable to everything from automating small sets of instructions, to building websites, to fully fledged applications. Jessica Hamrick, a psychology PhD student at the University of California, Berkeley, has been ▸

▸ programming in Python since 2008 and uses it in all phases of her research. In a study investigating how people manipulate geometric objects in their minds, for instance, she used the language (as well as JavaScript) to generate different shapes, present those to study participants, record their choices and analyse the data.

Despite its general-purpose power, Python is considered less painful for beginners to learn than other options. That accessibility is a function of both the language itself and the resources that have been built up around it (see 'A Python toolkit'). For example, software execution can be interactive — type a command, get a response — whereas in C, a compilation step is required to translate the code into an executable file, which complicates the process for neophytes. The language is also generally easier to handle; users do not have to predefine whether a variable will hold numbers or text, for instance. The classic programming exercise of printing 'Hello, world!' to the screen is as simple as it can be in Python — just type `print("Hello, world!")` at a Python prompt and hit Enter. "It's easier to teach novice programmers how to get things done in Python than in C++ or C," says Brown, now at the University of California, Davis. Python is in fact a popular choice for introductory programming classes in general.

The community aspect is particularly important to Python's growing adoption. Programming languages are popular only if new people are learning them and using them in diverse contexts, says Jessica McKellar, a software-engineering manager at the file-storage service Dropbox and a director of the Python Software Foundation, the non-profit organization that promotes and advances the language. That kind of use sets up a "virtuous cycle", McKellar says: new users extend the language into new areas, which in turn attracts still more users.

The community seems especially dedicated to encouraging women, Brown notes. There are numerous women-centric resources available, including workshops offered by the Hackbright Academy in San Francisco, the non-profit organization Ladies Learning Code in Toronto, Canada, and the global mentorship group PyLadies. As a master's student at McGill University in Montreal, Canada, Emily Irvine picked up Python to help her make sense of neuronal electrophysiology data. She was attracted to the language because of its "simple syntax" and "massive amount of online support". But just as important was the wider Python community, says Irvine, who will start a PhD in neuroscience at Dartmouth College in Hanover, New Hampshire, this autumn. At the PyCon conference last April in Montreal, "they just had such a welcoming atmosphere, especially towards women and scientists".

Educational resources also abound. The Software Carpentry Foundation runs a series of two-day workshops that focus on scientific programming, and many of its educational resources are available online. Online classes are also available through Coursera in Mountain View, California, and Edx in Cambridge, Massachusetts, as are do-it-yourself tutorials, such as those hosted by Codecademy in New York City. (Because Python is named in honour of Monty Python, these tutorials often work references to the British comedy troupe into their exercises: one Codecademy exercise, for example, is to capitalize and calculate the length of the phrase 'the ministry of silly walks'.)

A PYTHON TOOLKIT

How to get started

- Install Python through Anaconda or Enthought Canopy and find documentation at the Python Software Foundation
- Lessons for beginners can be found at Software Carpentry; Learn Python the Hard Way; Codecademy; and Think Python
- Other online resources on Python programming include a course from the Massachusetts Institute of Technology in Cambridge, lecture notes from Thomas Robitaille at the Max Planck Institute for Astronomy in Heidelberg, Germany, and a widely recommended essay from Google's head of research, Peter Norvig
- Open-source packages are available through SciPy.org
- Guides to programming and community support are available through Ladies Learning Code and Stack Overflow. PyCon.org lists conferences around the world.

Links to these resources can be found at go.nature.com/x2pzh1

Irvine taught herself to code using online courses and a healthy dose of the programming Q&A site stackoverflow.com. Today, she says, she considers herself somewhere between a beginner and an intermediate Python programmer, or 'pythonista', as they are sometimes called.

THE FULL MONTY

Of course, user-friendliness is meaningless if researchers cannot write the software they need. That is where Python's packages, which extend the language with new functionality, come into play. "Python was developed as a language with a philosophy that it was 'batteries included'," McKellar says — it has built-in capabilities that make it easy to get started right out of the box. But, "it also has a very mature package ecosystem around it. Anything that you could possibly write code to solve, people have written libraries to make that easier for you."

➲ NATURE.COM
For more on scientific software, apps and online tools, visit:
nature.com/toolbox

Scientific programmers, irrespective of their discipline, routinely use a small set of core packages: NumPy (mathematical arrays), SciPy (linear algebra, differential equations, signal processing and more), SymPy (symbolic mathematics), matplotlib (graph plotting) and Pandas (data analysis). Another popular tool, Cython, addresses Python's relatively slow execution speed. Cython optimizes certain aspects of Python code, such as 'for' loops (used to instruct a program to repeatedly run a specific block of code) that are notoriously slow, essentially by converting them into C. "You can get speed-ups that are up to 1,000 times faster than standard Python," says Paul Nation, a theoretical physicist at Korea University in Seoul.

The IPython Notebook is another popular package — Howe terms it "a coder's lab notebook" — that allows users to interleave data, code and explanatory text in a single browser-based page, rather than in separate files (see *Nature* 515, 151–152; 2014).

Beyond the core packages, software packages exist for just about every scientific discipline, including scikit-Learn for machine learning, Biopython for bioinformatics, PsychoPy for psychology and neuroscience and Astropy for astronomers. Thomas Robitaille, a coordinator of the Astropy project and a researcher at the Max Planck Institute for Astronomy in Heidelberg, Germany, says that Astropy was created to reduce duplicated effort between research groups. It gives users a core set of abilities, such as ways to convert coordinates from one astronomical mapping system to another, and a unified interface for reading and writing different data file formats, manipulating images and carrying out cosmological calculations. QuTip, another Python package, enables researchers working on quantum mechanics to define a system and then simulate how it behaves. The project was launched in 2010 by Nation and Robert Johansson, a postdoctoral fellow in RIKEN's Interdisciplinary Theoretical Science Research Group in Wako, Japan, to adapt into Python a MATLAB package that Nation was using.

Such packages are key enablers of McKellar's 'virtuous cycle'. But researchers could probably do their work using any language, provided they put in the time to learn it. (Indeed, in many languages, including Python, it is possible to run algorithms written in a different language, thereby allowing researchers to reuse their old code.) The difficult part of learning to program lies with the fundamentals, says Brown — once a researcher has those nailed down, adapting to a new language is just a matter of syntax. What matters most in the early stages is having a good support network. "Pick the programming language based on what people around you are using," Brown advises. Increasingly, that language is Python. ■

Jeffrey M. Perkel *is a writer based in Pocatello, Idaho.*

Chapter 4

Python: Invented for Productivity

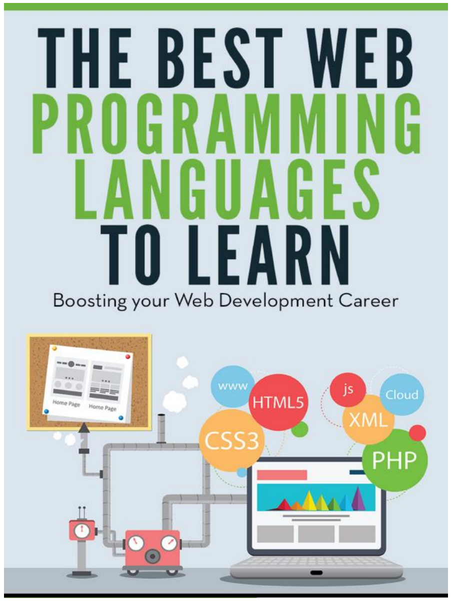
Python is such a general purpose langauge that is used in many applications and domains like web development, education, desktop GUI, software development etc. It sands for handling integration tasks, being versatile and extensible (the Python Packages include thousands of third party modules for python). In web development, Python is mostly found on the back end side with frameworks such as Django and Pyramid, micro-frameworks like Flask and Bottle and advanced content managment systems such as Plone.

4.1 Python Features

Python did not become so popular by chance. There stand several reasons (features) why this langauge made it to a lot of applications and users all around the world. The following introduce the most important ones:

- **Easy to Learn** - Nobody would prefer a language that is hard to learn (at least the basics). Python is a minimalistic language. It makes programmers focus on the problem and not on the language because of the pseudo-code it has that feels like rading English. The syntax is so simple that most people get surprised when being introduced to it for the first time.
- **Open Source** - To be clear, open source is not just about being free. It means people can freely distribute copies of it, access the source code and most importantly, make changes to it for use in personal projects. The language is constantly being imporved by a wide community of developers.
- **High-Level Language** - The higher the language level, the less you have to worry about how the machine understands it. For example, if in C you have to think of memory allocation, you don't have to in Python. Low level details are avoided in a lot of aspects in Python.
- **Portability** - Being open source comes with great advantages, and one of them is bringing the language to a whole bunch of platforms like Linux, Windows, Macintosh, FreeBSD etc. If you are careful enough to avoid system dependent code, than your programs wil work seamlessly in all platforms.
- **Object Oriented** - Both procedure-oriented programming (which is building a program around procedures/functions that are pieces of reusable code) and object-oriented programming are supported in Python. They are powerful tools in the hands of developers and come in really simple ways of implementation.
- **Extensibility** - Python takes future growth of implementation seriously. You can code a part of the program in `C` or `C++` for performance reasons and use it in your Python program. Python is also embeddable. You can embed it within your `C/C++` programs to give them "scripting" capabilites for your program's users.
- **Interpreted** - Python instructions are executed directly, without previously being compiled into machine-language instructions. In other words, it does not need compilation to binary code, and you can run the program directly from the source code.

Just to show you how clean Python code can be, just look at this function declaration:



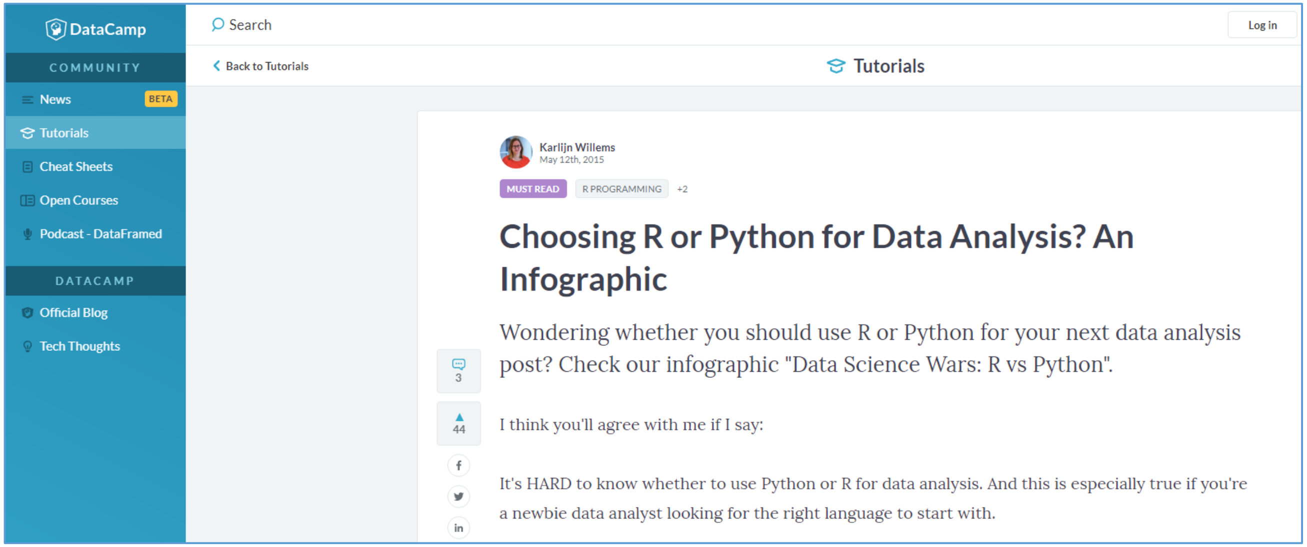
4.2 Powering Popular Sites

Some of the world's most popular sites use Python to make their services available to every user. Some of these websites include:

- **Google** - the most popular search engine uses Python to handle the traffic and computing needs of the search engine and its connected apps.
- **Youtube** - Python is used widely in Youtube to power all the features we love. Everytime you watch a video on Youtube, you are actually executing a bunch of Python code.
- **Dropbox** - uses a combination of wxPython and PyObjC on the Mac to deliver service bound to the disk and network.
- **Quora** - Adam d'Anggelo, answering to the choice of Python for Quora development, says it has only two drawbacks, speed and typechecking. He says Python was found to be fast enough for what they needed it, while they worked hard to reinvent some technology to solve typechecking.
- **Survey Monkey** - yet another service that started the application from the beginning to write it in Python, which was a rather promising language for testing and deploying new features.

Python was designed for web servers that handle a large amount of traffic, that is why it is chosen by industry leaders. It helps do more with fewer lines of readable code.

Check out some useful Python Tutorials



Karlijn Willems
May 12th, 2015

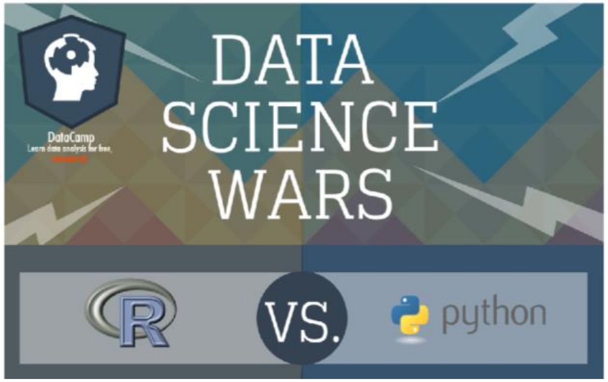
MUST READ R PROGRAMMING +2

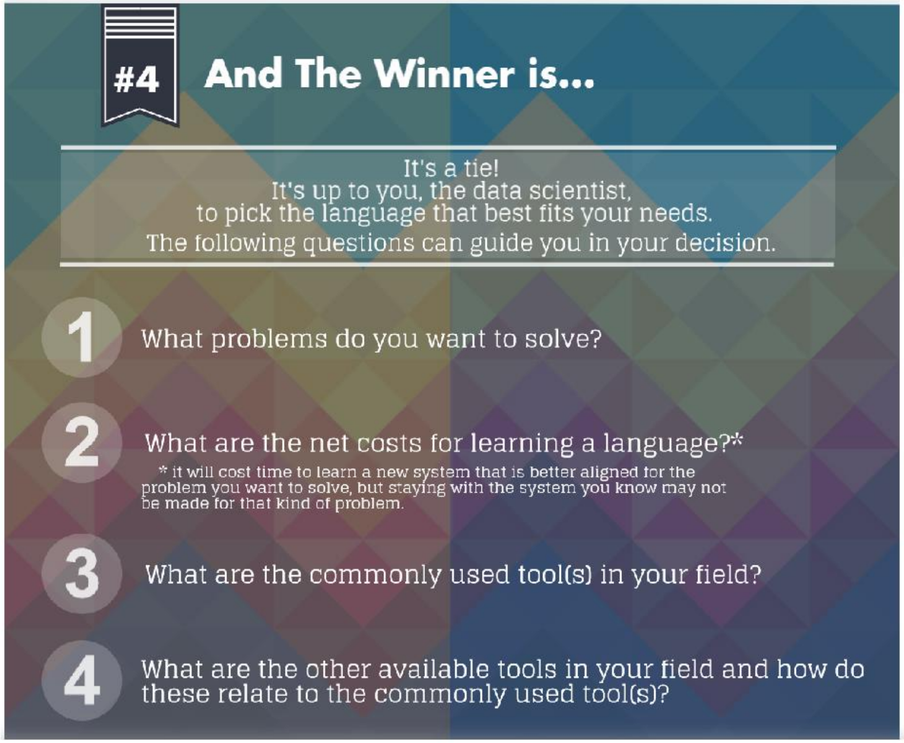
Choosing R or Python for Data Analysis? An Infographic

Wondering whether you should use R or Python for your next data analysis post? Check our infographic "Data Science Wars: R vs Python".

I think you'll agree with me if I say:

It's HARD to know whether to use Python or R for data analysis. And this is especially true if you're a newbie data analyst looking for the right language to start with.





#4 And The Winner is...

It's a tie!
It's up to you, the data scientist,
to pick the language that best fits your needs.
The following questions can guide you in your decision.

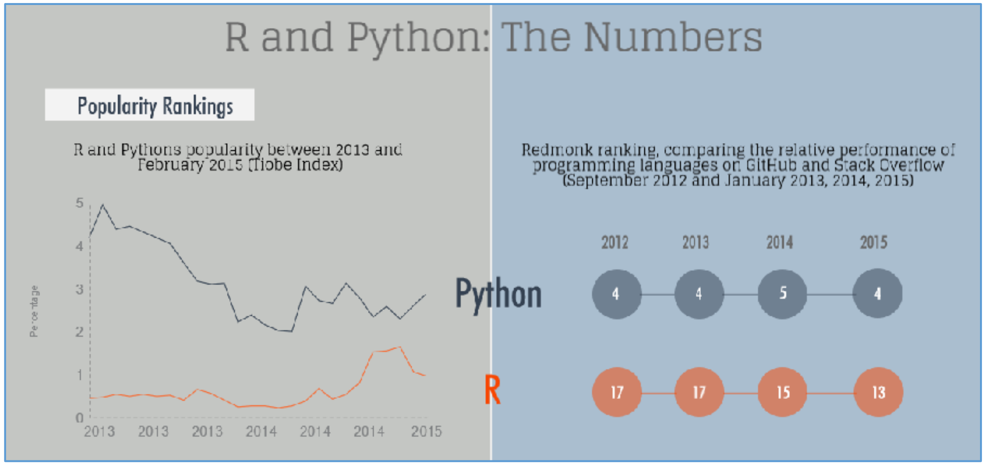
1 What problems do you want to solve?

2 What are the net costs for learning a language?*

* it will cost time to learn a new system that is better aligned for the problem you want to solve, but staying with the system you know may not be made for that kind of problem.

3 What are the commonly used tool(s) in your field?

4 What are the other available tools in your field and how do these relate to the commonly used tool(s)?

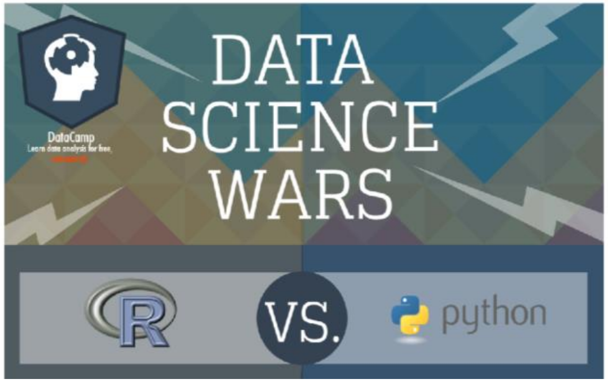


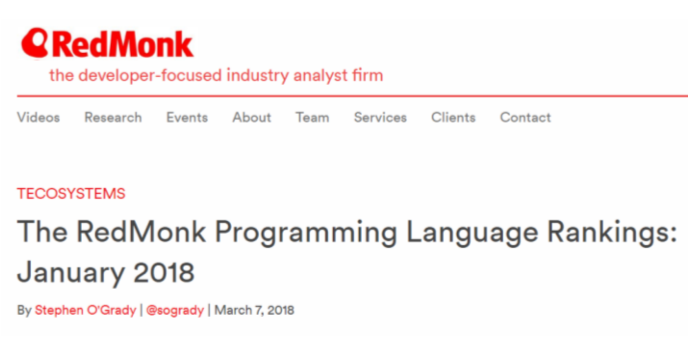
R and Python: The Numbers

Popularity Rankings

R and Pythons popularity between 2013 and February 2015 (Tiobe Index)

Redmonk ranking, comparing the relative performance of programming languages on GitHub and Stack Overflow (September 2012 and January 2013, 2014, 2015)





RedMonk

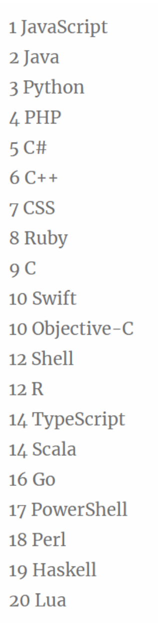
the developer-focused industry analyst firm

Videos Research Events About Team Services Clients Contact

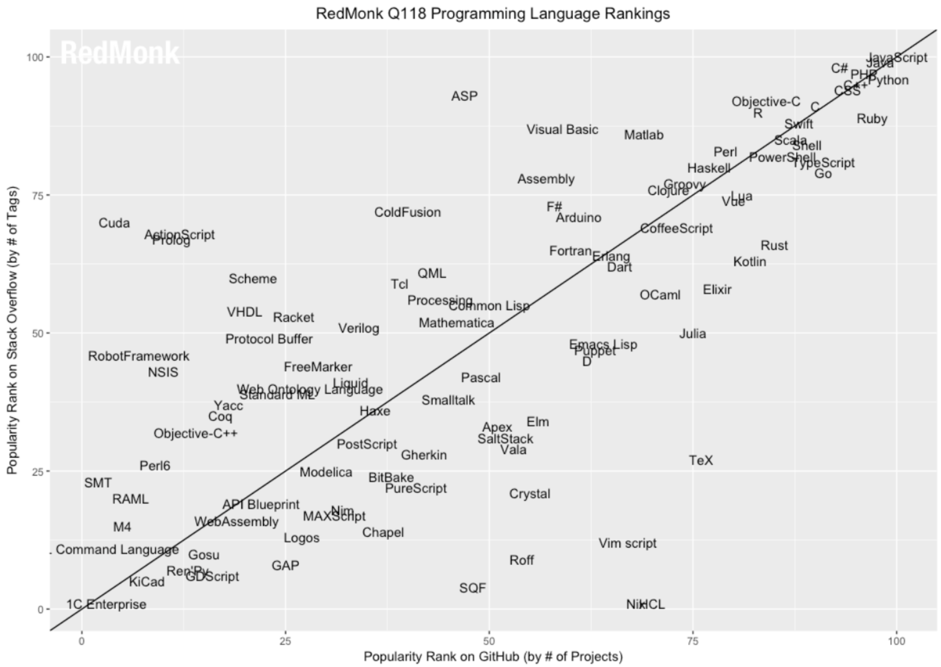
TECOSYSTEMS

The RedMonk Programming Language Rankings: January 2018

By Stephen O'Grady | @sogrady | March 7, 2018



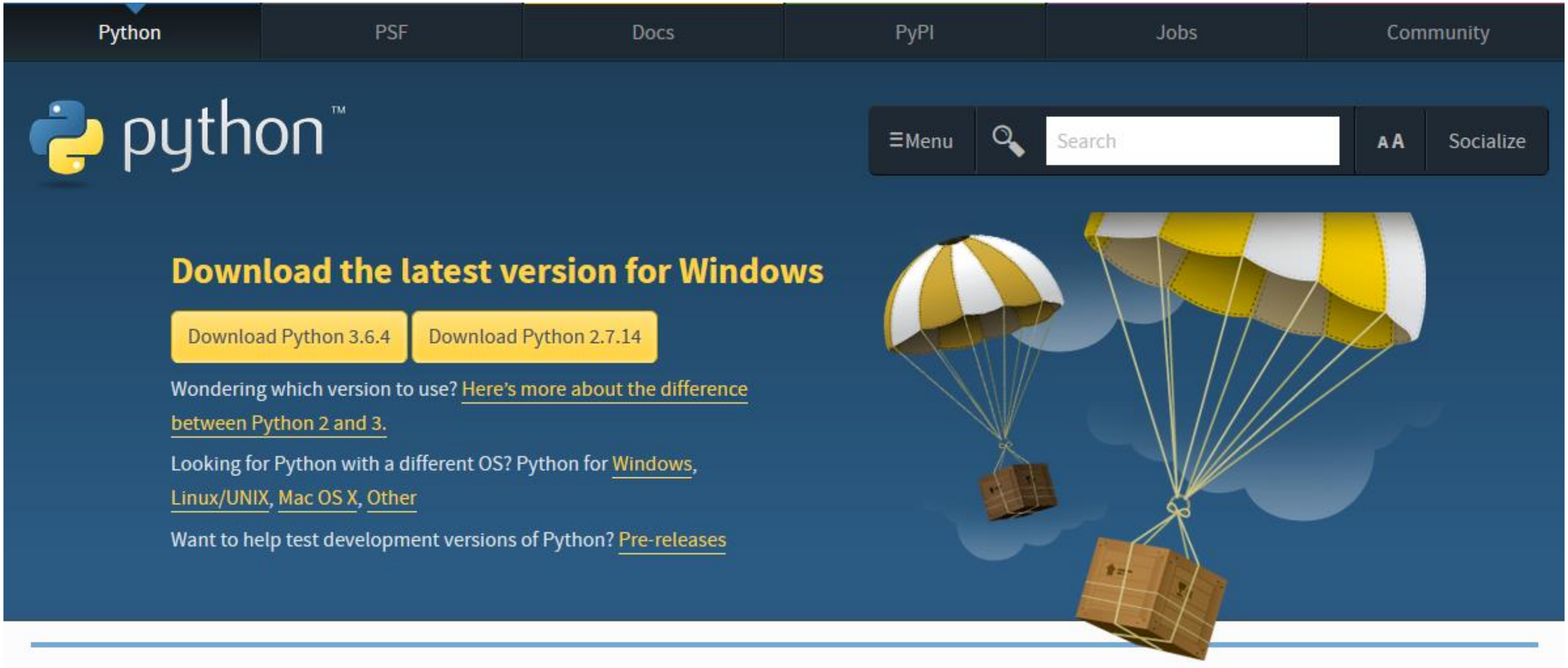
1 JavaScript
2 Java
3 Python
4 PHP
5 C#
6 C++
7 CSS
8 Ruby
9 C
10 Swift
10 Objective-C
12 Shell
12 R
14 TypeScript
14 Scala
16 Go
17 PowerShell
18 Perl
19 Haskell
20 Lua



RedMonk Q118 Programming Language Rankings

Kingston University London

Installation



Download the latest version for Windows

Download Python 3.6.4 | Download Python 2.7.14

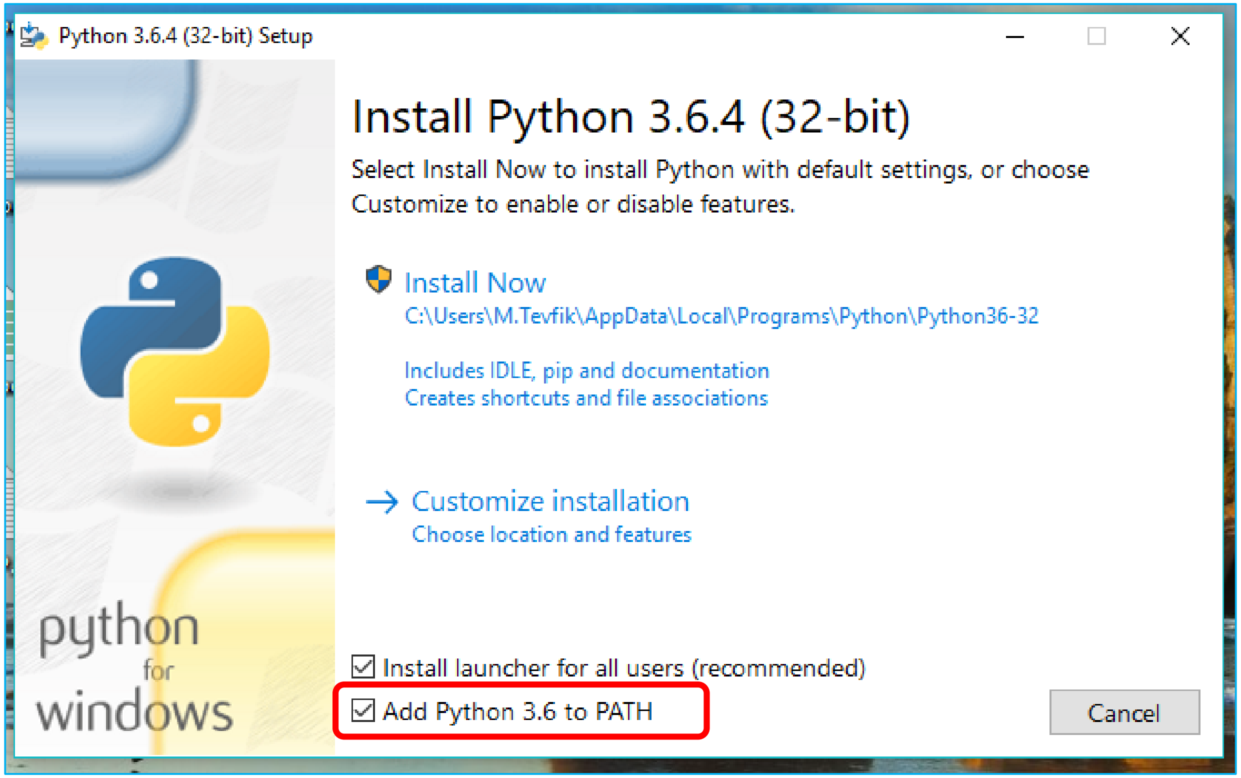
Wondering which version to use? Here's more about the difference between Python 2 and 3.

Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Pre-releases

Kingston University London

Installation



Python 3.6.4 (32-bit) Setup

Install Python 3.6.4 (32-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\M.Tevfik\AppData\Local\Programs\Python\Python36-32

Includes IDLE, pip and documentation
Creates shortcuts and file associations

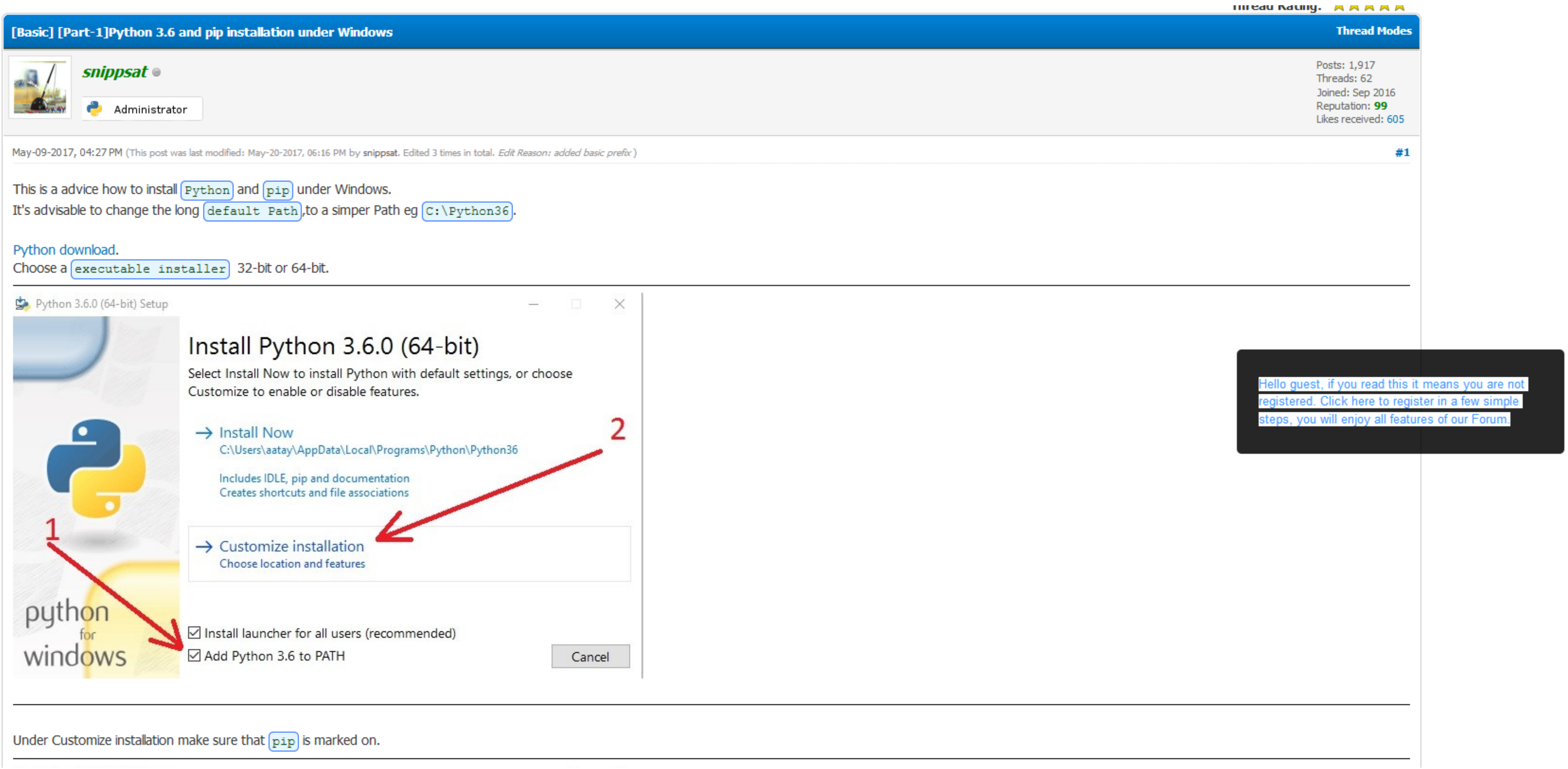
→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)

☑ Add Python 3.6 to PATH

Installation

Follow this tutorial to the letter!



Thread Rating: ★★★★★

[Basic] [Part-1]Python 3.6 and pip installation under Windows

Thread Modes

snippsat

Administrator

Posts: 1,917
Threads: 62
Joined: Sep 2016
Reputation: 99
Likes received: 605

May-09-2017, 04:27 PM (This post was last modified: May-20-2017, 06:16 PM by snippsat. Edited 3 times in total. *Edit Reason: added basic prefix*)

#1

This is a advice how to install `Python` and `pip` under Windows.
It's advisable to change the long `default Path`,to a simper Path eg `C:\Python36`.

Python download.
Choose a `executable installer` 32-bit or 64-bit.

Python 3.6.0 (64-bit) Setup

Install Python 3.6.0 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

→ Install Now
C:\Users\aatay\AppData\Local\Programs\Python\Python36

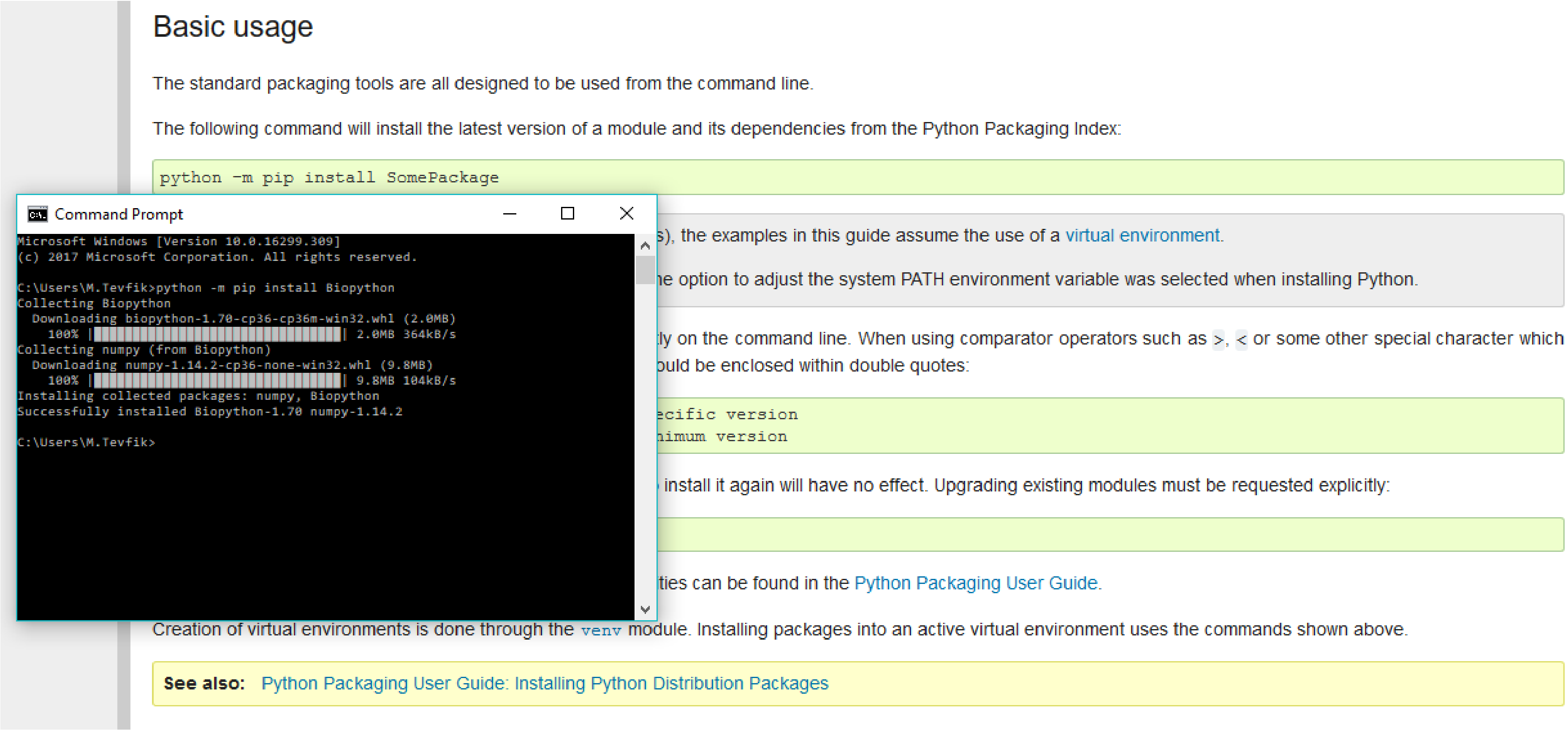
Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.6 to PATH

Under Customize installation make sure that `pip` is marked on.

Installation: Packages



Basic usage

The standard packaging tools are all designed to be used from the command line.

The following command will install the latest version of a module and its dependencies from the Python Packaging Index:

```
python -m pip install SomePackage
```

s), the examples in this guide assume the use of a virtual environment.

e option to adjust the system PATH environment variable was selected when installing Python.

ly on the command line. When using comparator operators such as >, < or some other special character which
ould be enclosed within double quotes:

```
ecific version
imum version
```

install it again will have no effect. Upgrading existing modules must be requested explicitly:

ties can be found in the Python Packaging User Guide.

Creation of virtual environments is done through the `venv` module. Installing packages into an active virtual environment uses the commands shown above.

See also: Python Packaging User Guide: Installing Python Distribution Packages

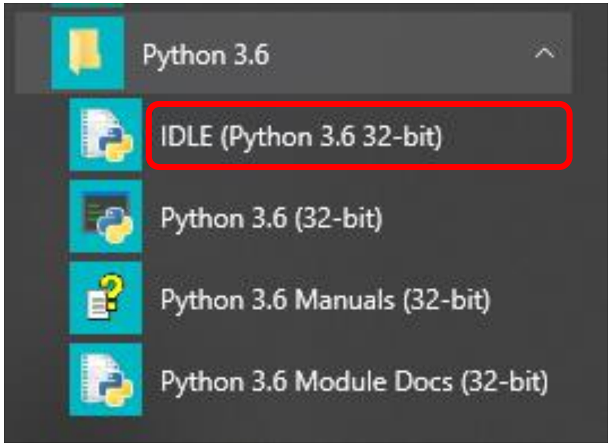
Command Prompt

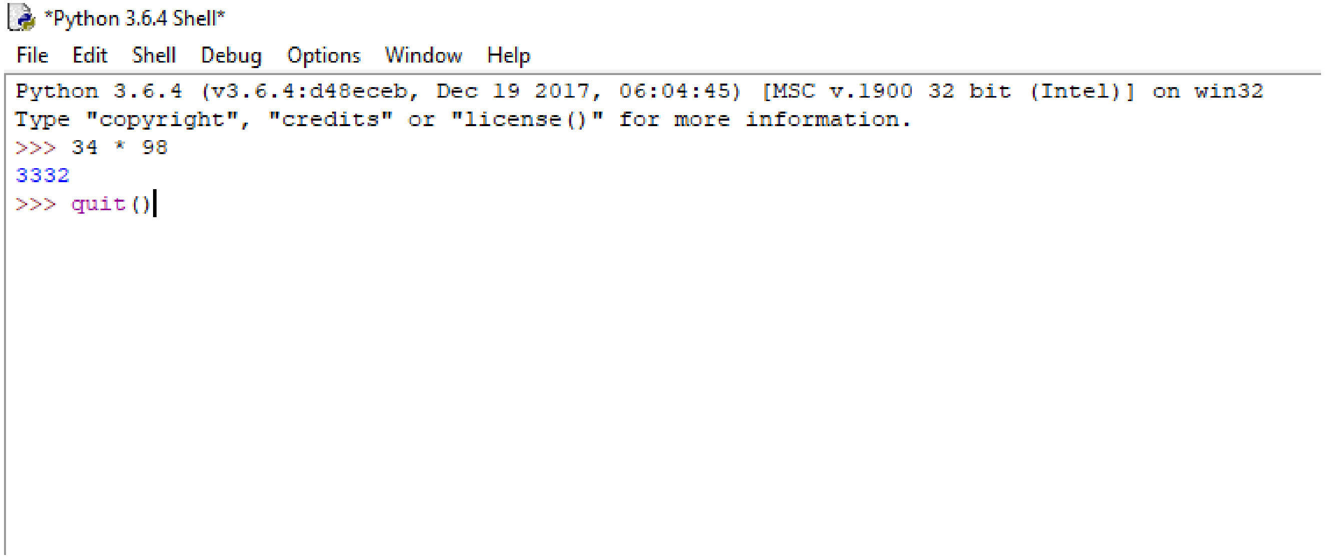
```
Microsoft Windows [Version 10.0.16299.309]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\M.Tevfik>python -m pip install Biopython
Collecting Biopython
  Downloading biopython-1.70-cp36-cp36m-win32.whl (2.0MB)
    100% |████████████████████████████████| 2.0MB 364kB/s
Collecting numpy (from Biopython)
  Downloading numpy-1.14.2-cp36-none-win32.whl (9.8MB)
    100% |████████████████████████████████| 9.8MB 104kB/s
Installing collected packages: numpy, Biopython
Successfully installed Biopython-1.70 numpy-1.14.2

C:\Users\M.Tevfik>
```

Do this on command line, not while in Python
(use `exit()` to make sure you have exited Python)

Kingston
University
London

IDLE

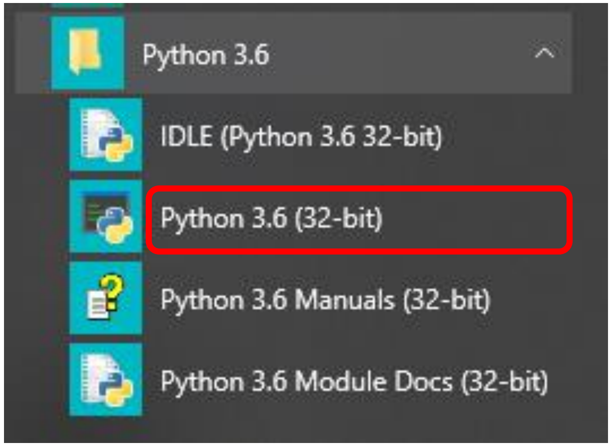


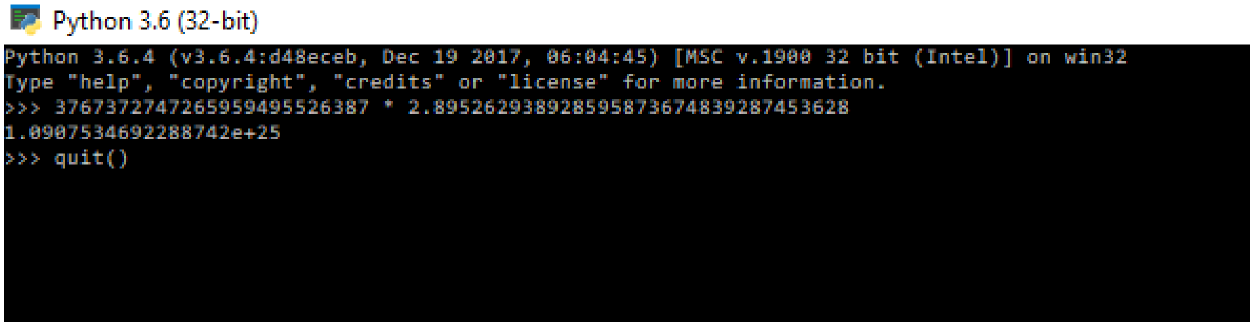


```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 34 * 98
3332
>>> quit()
```

Kingston University London

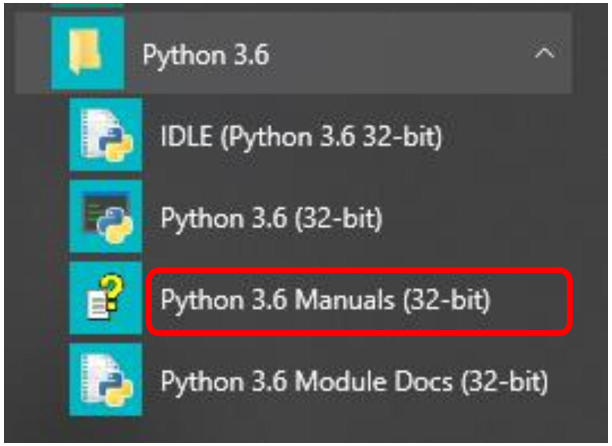
Python

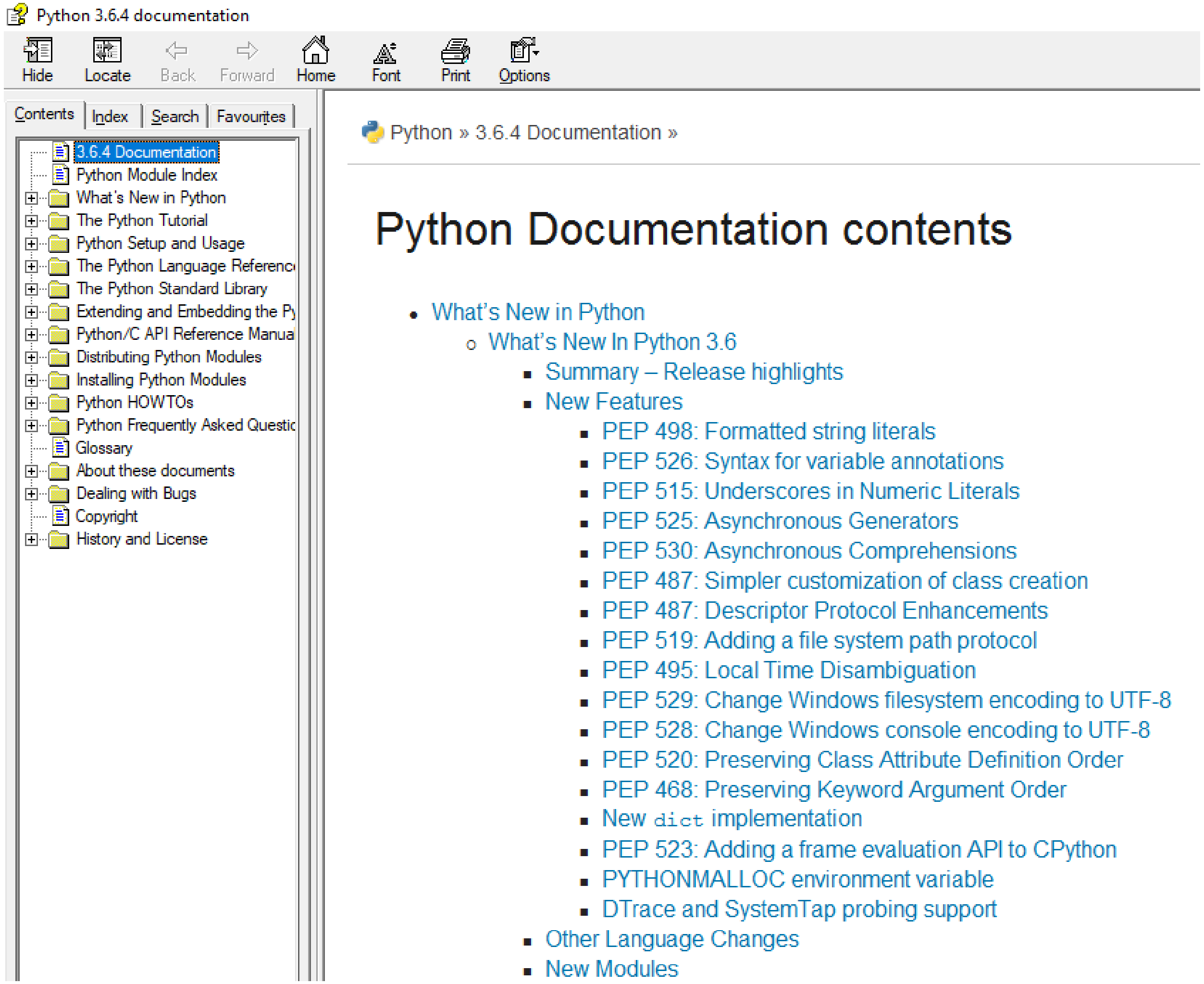




```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 3767372747265959495526387 * 2.895262938928595873674839287453628
1.0907534692288742e+25
>>> quit()
```

Manual





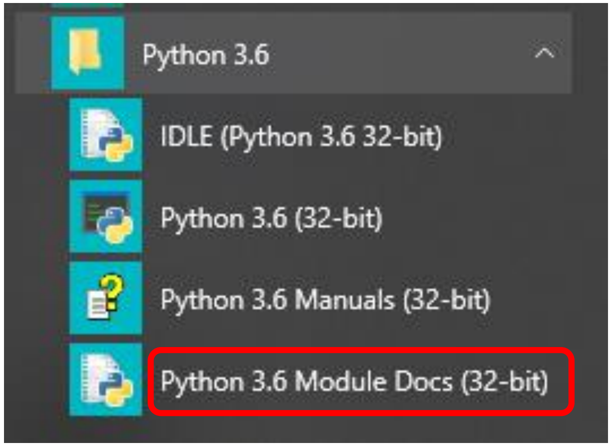
- 3.6.4 Documentation
- Python Module Index
- What's New in Python
- The Python Tutorial
- Python Setup and Usage
- The Python Language Referenc
- The Python Standard Library
- Extending and Embedding the Py
- Python/C API Reference Manua
- Distributing Python Modules
- Installing Python Modules
- Python HOWTOs
- Python Frequently Asked Questic
- Glossary
- About these documents
- Dealing with Bugs
- Copyright
- History and License

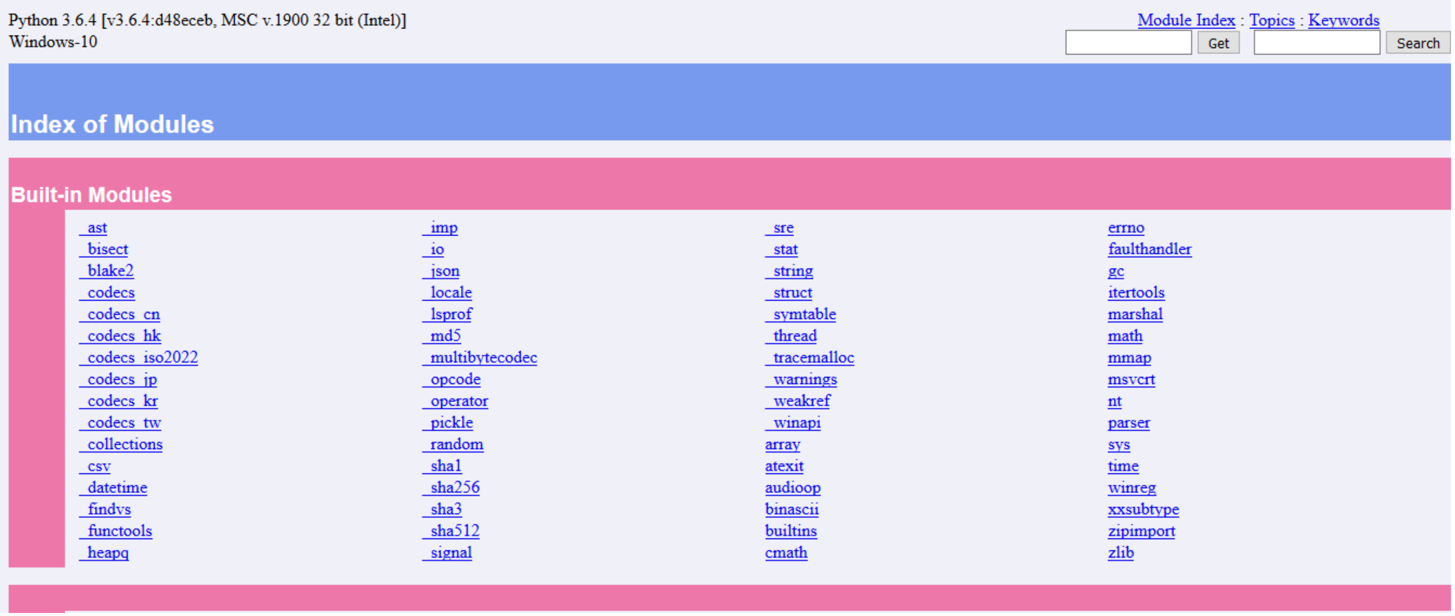
Python » 3.6.4 Documentation »

Python Documentation contents

- What's New in Python
  - What's New In Python 3.6
    - Summary – Release highlights
    - New Features
      - PEP 498: Formatted string literals
      - PEP 526: Syntax for variable annotations
      - PEP 515: Underscores in Numeric Literals
      - PEP 525: Asynchronous Generators
      - PEP 530: Asynchronous Comprehensions
      - PEP 487: Simpler customization of class creation
      - PEP 487: Descriptor Protocol Enhancements
      - PEP 519: Adding a file system path protocol
      - PEP 495: Local Time Disambiguation
      - PEP 529: Change Windows filesystem encoding to UTF-8
      - PEP 528: Change Windows console encoding to UTF-8
      - PEP 520: Preserving Class Attribute Definition Order
      - PEP 468: Preserving Keyword Argument Order
      - New `dict` implementation
      - PEP 523: Adding a frame evaluation API to CPython
      - PYTHONMALLOC environment variable
      - DTrace and SystemTap probing support
    - Other Language Changes
    - New Modules

Module Docs




Python 3.6.4 [v3.6.4:d48eceb, MSC v.1900 32 bit (Intel)]
Windows-10

Module Index : Topics : Keywords

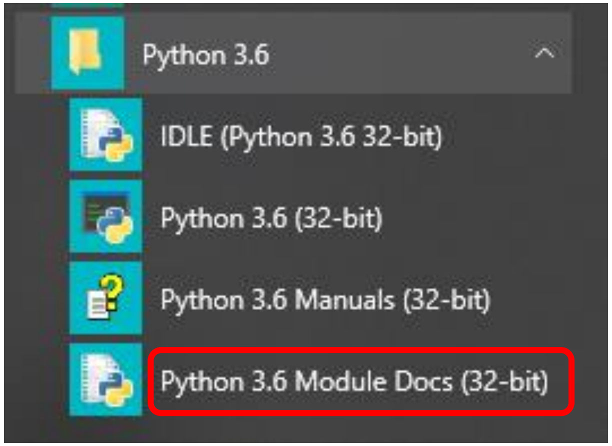
Get Search

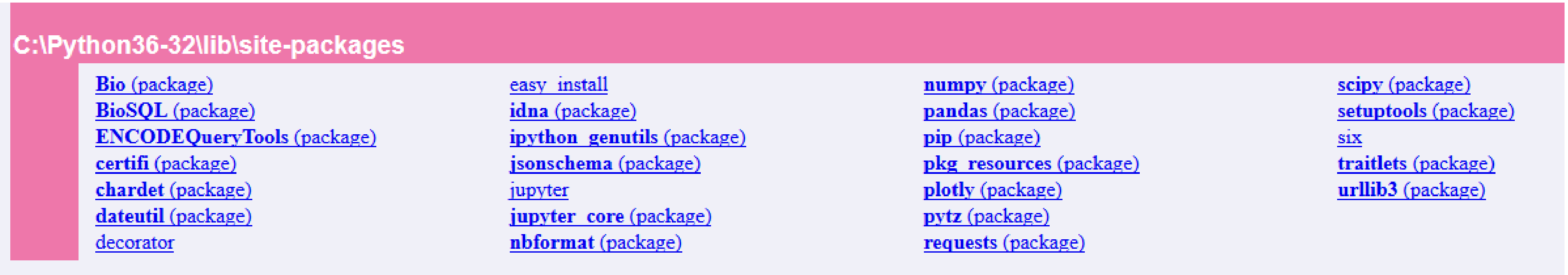
Index of Modules

Built-in Modules

- _ast
- _bisect
- _blake2
- _codecs
- _codecs_cn
- _codecs_hk
- _codecs_iso2022
- _codecs_jp
- _codecs_kr
- _codecs_tw
- _collections
- _csv
- _datetime
- _findvs
- _functools
- _heapq
- _imp
- _io
- _json
- _locale
- _lsprof
- _md5
- _multibytecodec
- _opcode
- _operator
- _pickle
- _random
- _sha1
- _sha256
- _sha3
- _sha512
- _signal
- _sre
- _stat
- _string
- _struct
- _symtable
- _thread
- _tracemalloc
- _warnings
- _weakref
- _winapi
- array
- atexit
- audioop
- binascii
- builtins
- cmath
- errno
- faulthandler
- gc
- itertools
- marshal
- math
- mmap
- msvcrt
- nt
- parser
- sys
- time
- winreg
- xxsubtype
- zipimport
- zlib

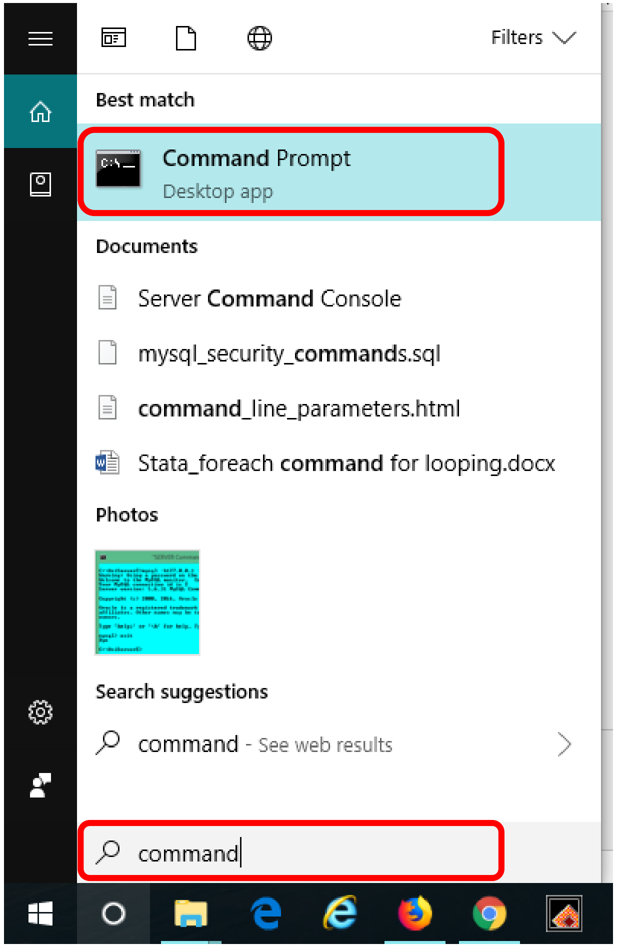
Module Docs



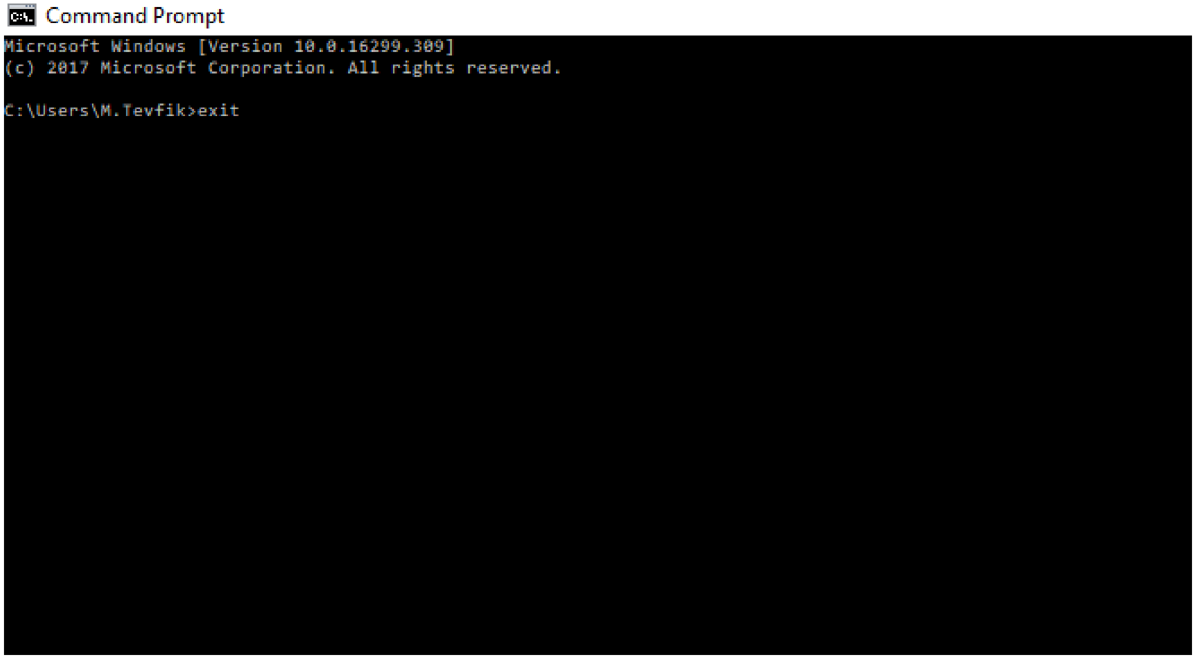

C:\Python36-32\lib\site-packages

- **Bio** (package)
- **BioSQL** (package)
- **ENCODEQueryTools** (package)
- **certifi** (package)
- **chardet** (package)
- **dateutil** (package)
- decorator
- easy_install
- **idna** (package)
- **ipython_genutils** (package)
- **jsonschema** (package)
- jupyter
- **jupyter_core** (package)
- **nbformat** (package)
- **numpy** (package)
- **pandas** (package)
- **pip** (package)
- **pkg_resources** (package)
- **plotly** (package)
- **pytz** (package)
- **requests** (package)
- **scipy** (package)
- **setuptools** (package)
- six
- **traitlets** (package)
- **urllib3** (package)

Command Prompt/Line


Filters
Best match
Command Prompt
Desktop app
Documents
Server Command Console
mysql_security_commands.sql
command_line_parameters.html
Stata_foreach command for looping.docx
Photos
Search suggestions
command - See web results
command

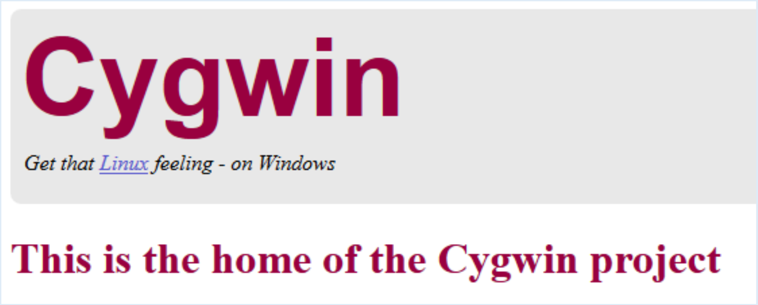
Windows equivalent of TERMINAL (Unix) or BASH (Linux)



```
Microsoft Windows [Version 10.0.16299.309]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\M.Tevfik>exit
```

Kingston
University
London

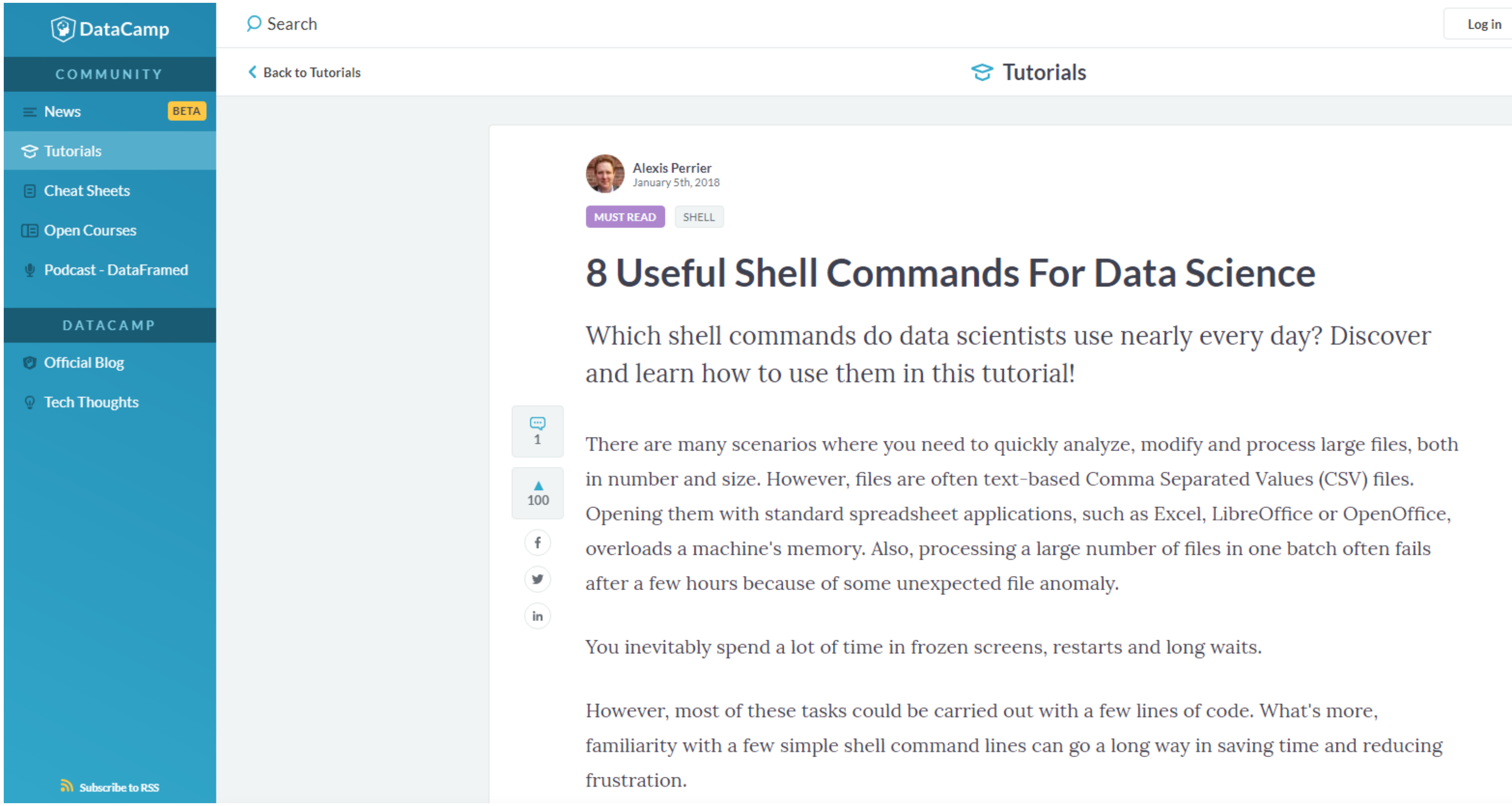
Linux Shell in Windows



Cygwin

Get that Linux feeling - on Windows

This is the home of the Cygwin project



Alexis Perrier
January 5th, 2018

MUST READ SHELL

8 Useful Shell Commands For Data Science

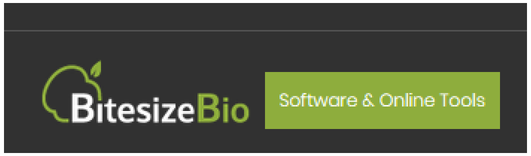
Which shell commands do data scientists use nearly every day? Discover and learn how to use them in this tutorial!

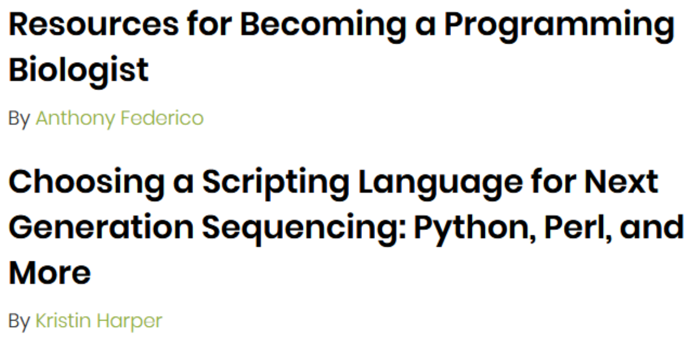
There are many scenarios where you need to quickly analyze, modify and process large files, both in number and size. However, files are often text-based Comma Separated Values (CSV) files. Opening them with standard spreadsheet applications, such as Excel, LibreOffice or OpenOffice, overloads a machine's memory. Also, processing a large number of files in one batch often fails after a few hours because of some unexpected file anomaly.

You inevitably spend a lot of time in frozen screens, restarts and long waits.

However, most of these tasks could be carried out with a few lines of code. What's more, familiarity with a few simple shell command lines can go a long way in saving time and reducing frustration.

Introductory Reading



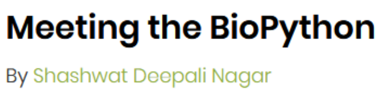


Resources for Becoming a Programming Biologist

By Anthony Federico

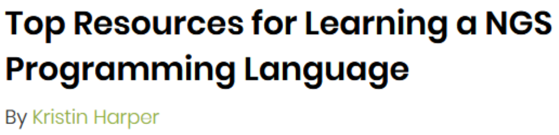
Choosing a Scripting Language for Next Generation Sequencing: Python, Perl, and More

By Kristin Harper



Meeting the BioPython

By Shashwat Deepali Nagar

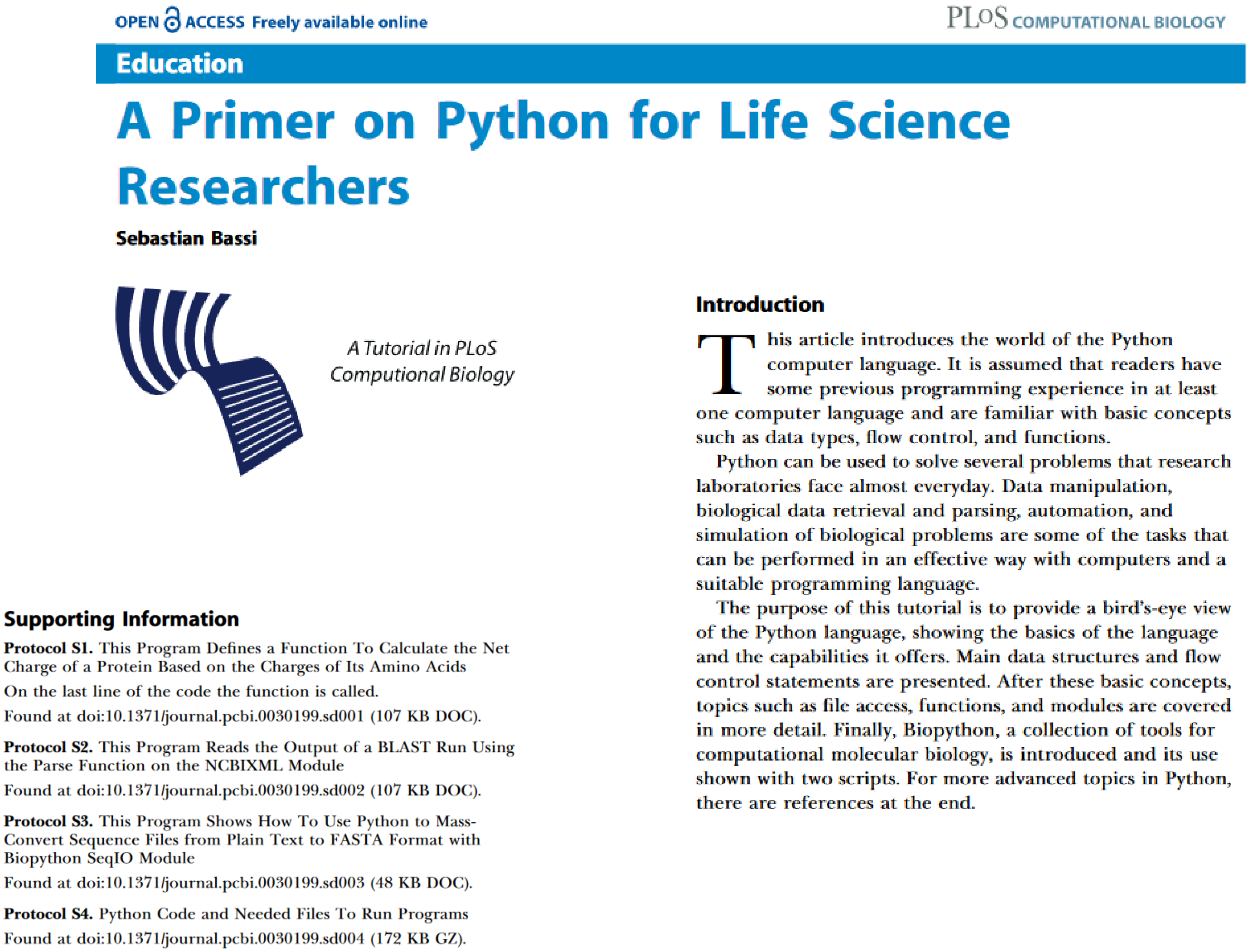


Top Resources for Learning a NGS Programming Language

By Kristin Harper

Python is one of the easiest programming languages to learn for biologists.

How to Begin



OPEN ACCESS Freely available online

PLoS COMPUTATIONAL BIOLOGY

Education

A Primer on Python for Life Science Researchers

Sebastian Bassi

A Tutorial in PLoS Computional Biology

Introduction

This article introduces the world of the Python computer language. It is assumed that readers have some previous programming experience in at least one computer language and are familiar with basic concepts such as data types, flow control, and functions.

Python can be used to solve several problems that research laboratories face almost everyday. Data manipulation, biological data retrieval and parsing, automation, and simulation of biological problems are some of the tasks that can be performed in an effective way with computers and a suitable programming language.

The purpose of this tutorial is to provide a bird's-eye view of the Python language, showing the basics of the language and the capabilities it offers. Main data structures and flow control statements are presented. After these basic concepts, topics such as file access, functions, and modules are covered in more detail. Finally, Biopython, a collection of tools for computational molecular biology, is introduced and its use shown with two scripts. For more advanced topics in Python, there are references at the end.

Supporting Information

Protocol S1. This Program Defines a Function To Calculate the Net Charge of a Protein Based on the Charges of Its Amino Acids

On the last line of the code the function is called.

Found at doi:10.1371/journal.pcbi.0030199.sd001 (107 KB DOC).

Protocol S2. This Program Reads the Output of a BLAST Run Using the Parse Function on the NCBIXML Module

Found at doi:10.1371/journal.pcbi.0030199.sd002 (107 KB DOC).

Protocol S3. This Program Shows How To Use Python to Mass-Convert Sequence Files from Plain Text to FASTA Format with Biopython SeqIO Module

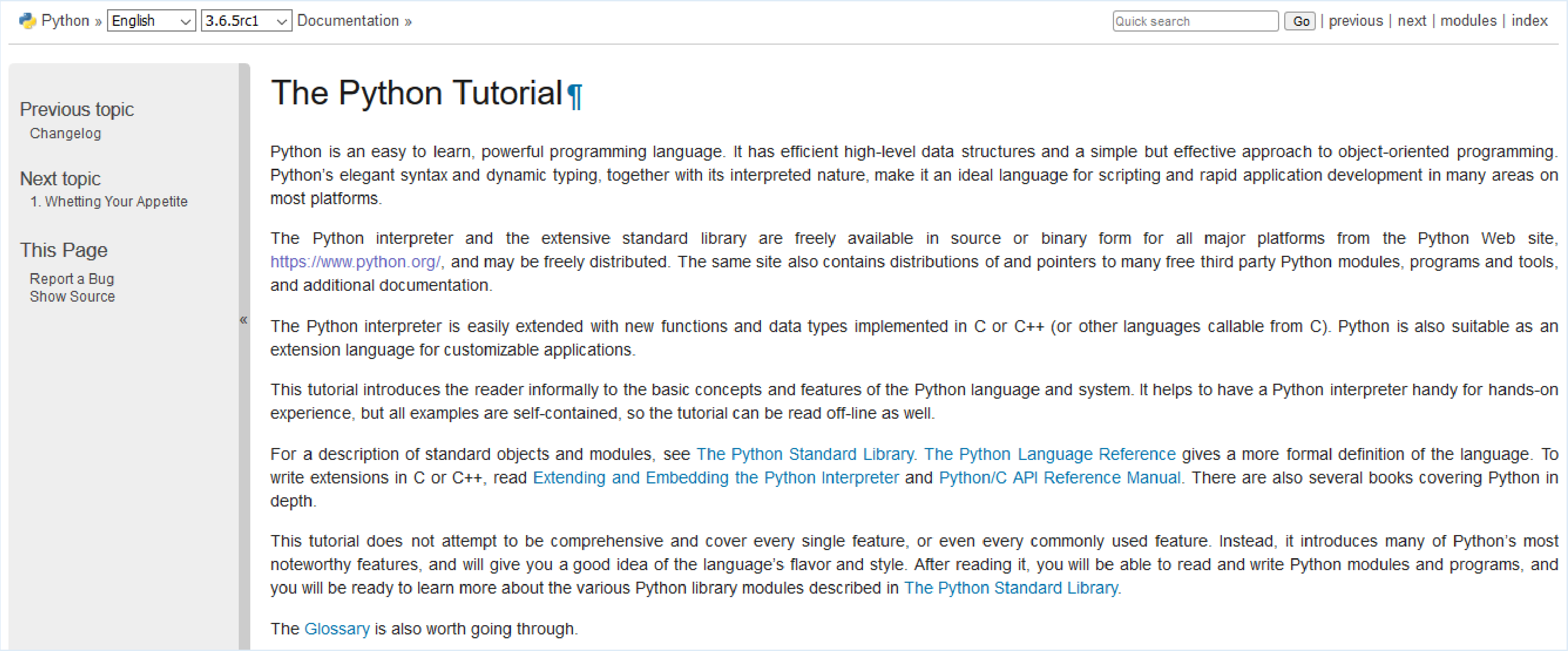
Found at doi:10.1371/journal.pcbi.0030199.sd003 (48 KB DOC).

Protocol S4. Python Code and Needed Files To Run Programs

Found at doi:10.1371/journal.pcbi.0030199.sd004 (172 KB GZ).

2007; Python 2!

Tutorial at Python Website


Python » English 3.6.5rc1 Documentation »
Quick search
Go
previous | next | modules | index
Previous topic
Changelog
Next topic
1. Whetting Your Appetite
This Page
Report a Bug
Show Source
«

The Python Tutorial¶

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, https://www.python.org/, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

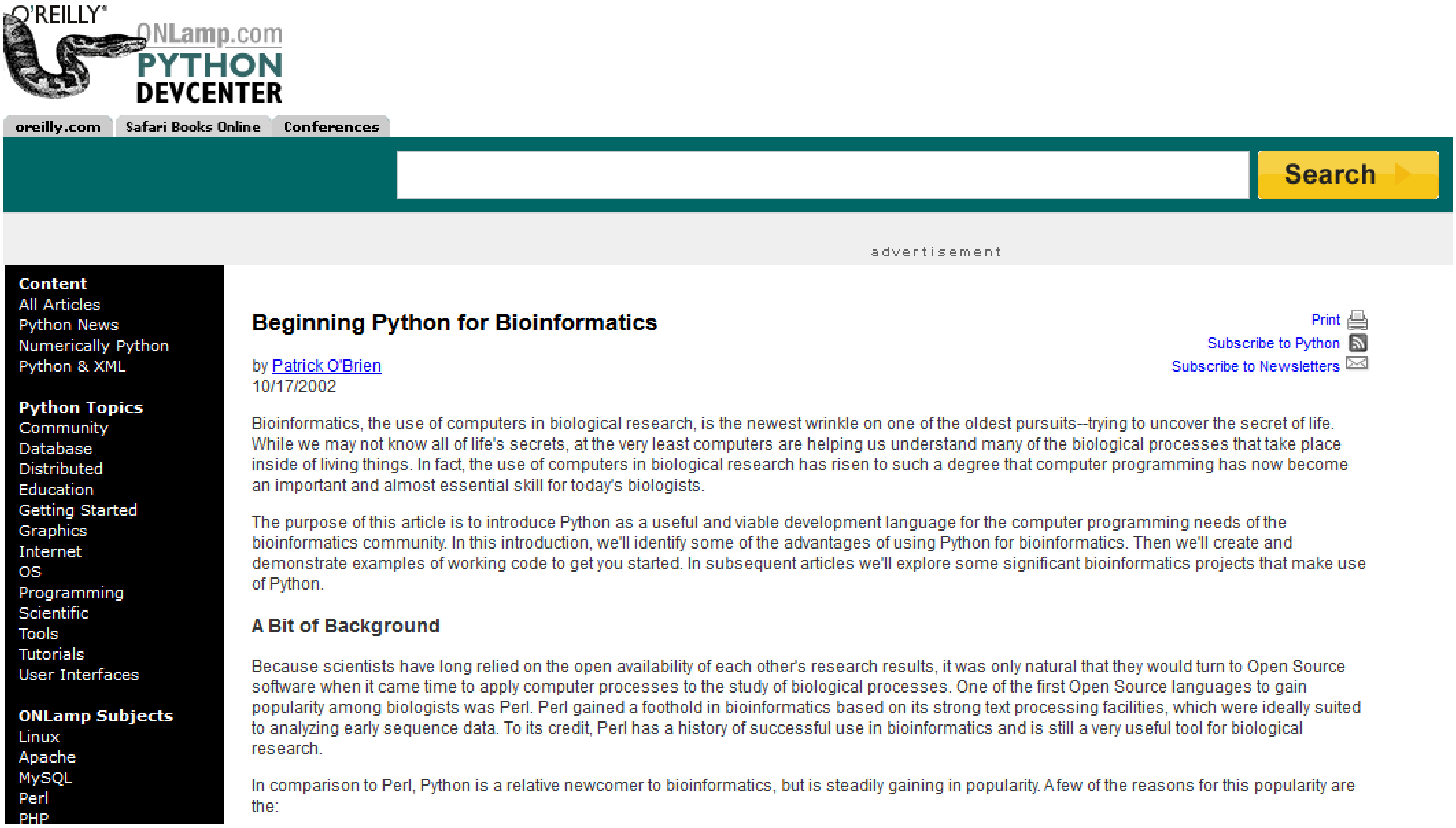
This tutorial introduces the reader informally to the basic concepts and features of the Python language and system. It helps to have a Python interpreter handy for hands-on experience, but all examples are self-contained, so the tutorial can be read off-line as well.

For a description of standard objects and modules, see The Python Standard Library. The Python Language Reference gives a more formal definition of the language. To write extensions in C or C++, read Extending and Embedding the Python Interpreter and Python/C API Reference Manual. There are also several books covering Python in depth.

This tutorial does not attempt to be comprehensive and cover every single feature, or even every commonly used feature. Instead, it introduces many of Python's most noteworthy features, and will give you a good idea of the language's flavor and style. After reading it, you will be able to read and write Python modules and programs, and you will be ready to learn more about the various Python library modules described in The Python Standard Library.

The Glossary is also worth going through.

Python & Bioinformatics



Beginning Python for Bioinformatics

by Patrick O'Brien
10/17/2002

Bioinformatics, the use of computers in biological research, is the newest wrinkle on one of the oldest pursuits--trying to uncover the secret of life. While we may not know all of life's secrets, at the very least computers are helping us understand many of the biological processes that take place inside of living things. In fact, the use of computers in biological research has risen to such a degree that computer programming has now become an important and almost essential skill for today's biologists.

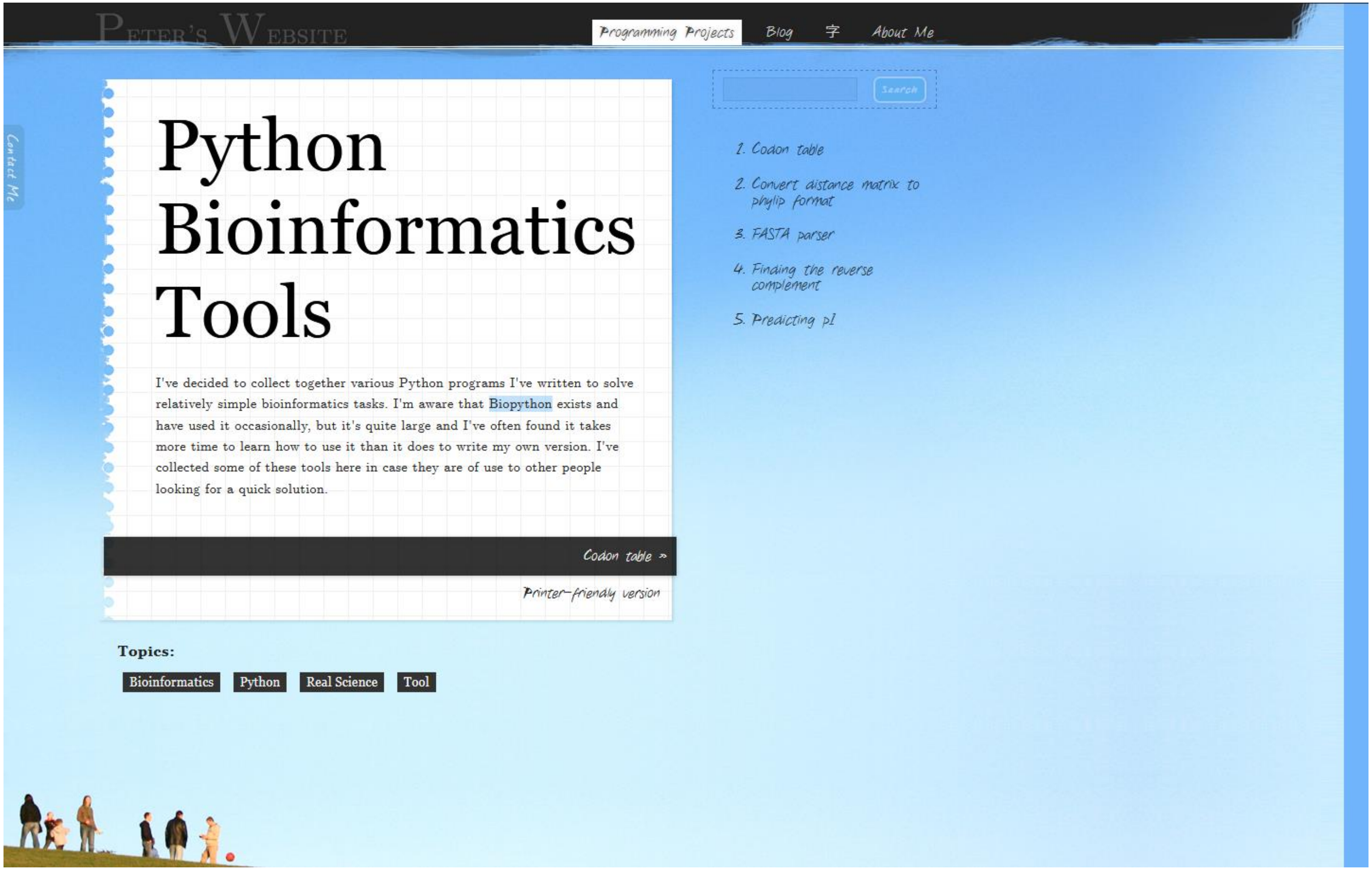
The purpose of this article is to introduce Python as a useful and viable development language for the computer programming needs of the bioinformatics community. In this introduction, we'll identify some of the advantages of using Python for bioinformatics. Then we'll create and demonstrate examples of working code to get you started. In subsequent articles we'll explore some significant bioinformatics projects that make use of Python.

A Bit of Background

Because scientists have long relied on the open availability of each other's research results, it was only natural that they would turn to Open Source software when it came time to apply computer processes to the study of biological processes. One of the first Open Source languages to gain popularity among biologists was Perl. Perl gained a foothold in bioinformatics based on its strong text processing facilities, which were ideally suited to analyzing early sequence data. To its credit, Perl has a history of successful use in bioinformatics and is still a very useful tool for biological research.

In comparison to Perl, Python is a relative newcomer to bioinformatics, but is steadily gaining in popularity. A few of the reasons for this popularity are the:

Python & Bioinformatics



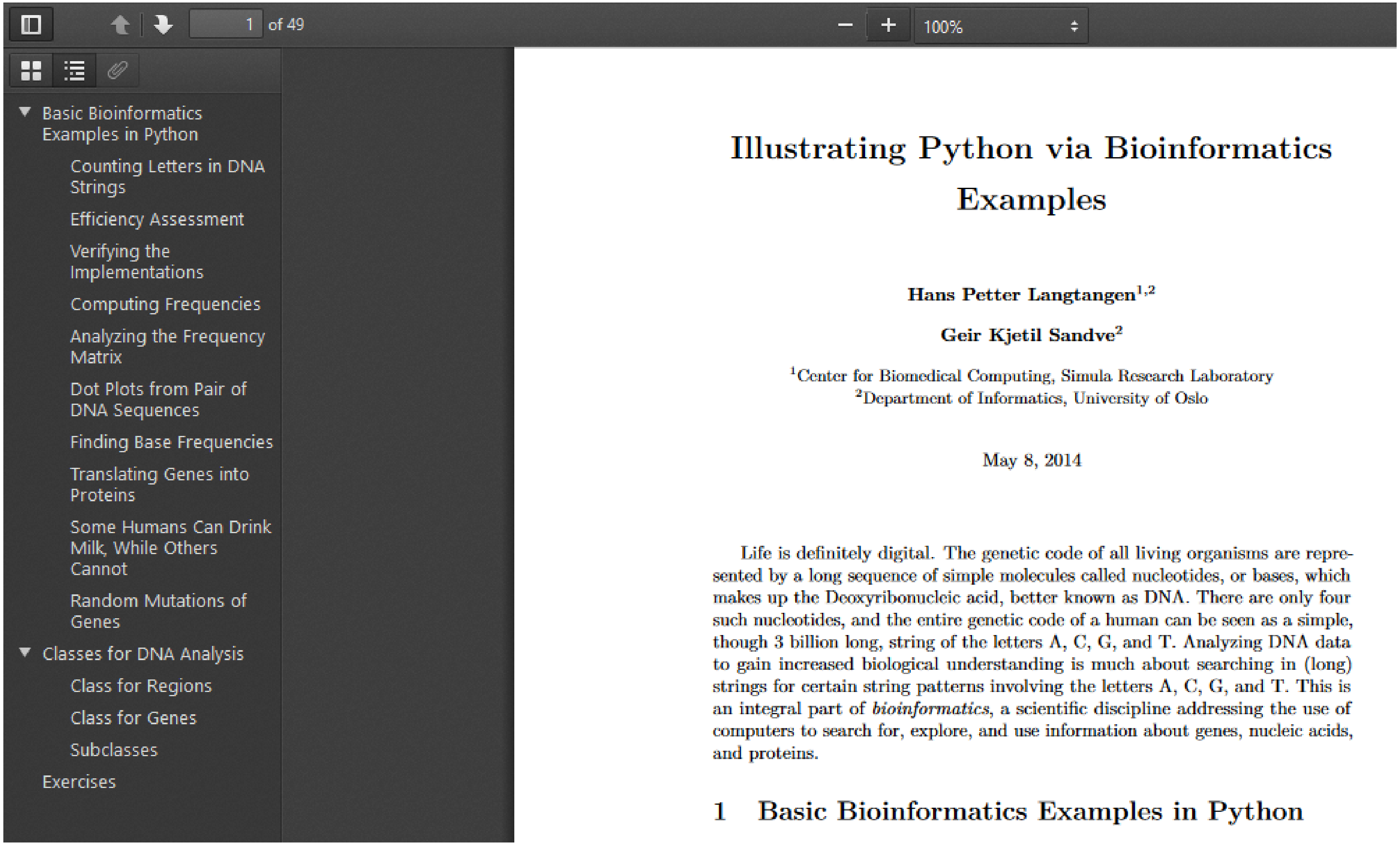
1. Codon table
2. Convert distance matrix to phylip format
3. FASTA parser
4. Finding the reverse complement
5. Predicting pI

Python Bioinformatics Tools

I've decided to collect together various Python programs I've written to solve relatively simple bioinformatics tasks. I'm aware that Biopython exists and have used it occasionally, but it's quite large and I've often found it takes more time to learn how to use it than it does to write my own version. I've collected some of these tools here in case they are of use to other people looking for a quick solution.

Topics:

Python & Bioinformatics


1
of 49
100%
Basic Bioinformatics Examples in Python
Counting Letters in DNA Strings
Efficiency Assessment
Verifying the Implementations
Computing Frequencies
Analyzing the Frequency Matrix
Dot Plots from Pair of DNA Sequences
Finding Base Frequencies
Translating Genes into Proteins
Some Humans Can Drink Milk, While Others Cannot
Random Mutations of Genes
Classes for DNA Analysis
Class for Regions
Class for Genes
Subclasses
Exercises

Illustrating Python via Bioinformatics Examples

Hans Petter Langtangen[1,2]

Geir Kjetil Sandve[2]

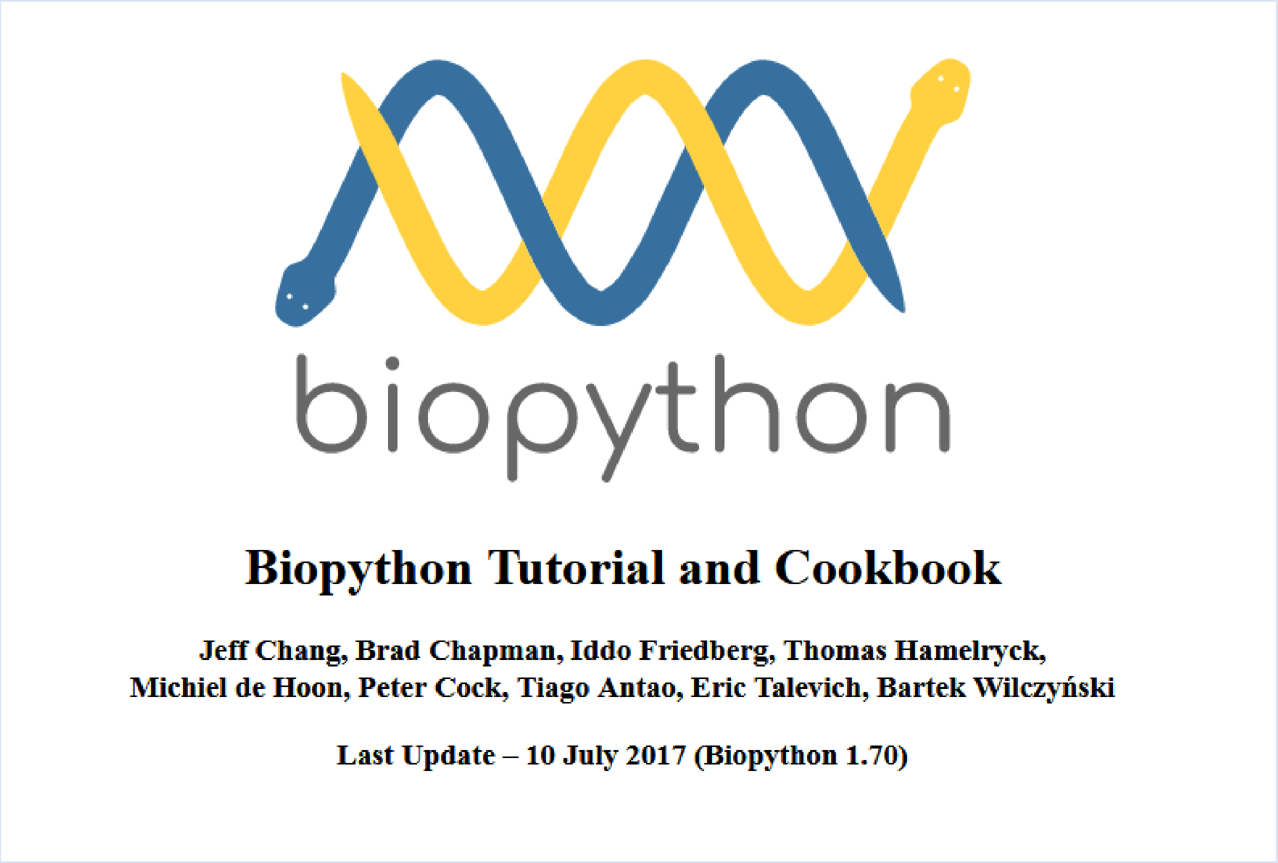
[1]Center for Biomedical Computing, Simula Research Laboratory
[2]Department of Informatics, University of Oslo

May 8, 2014

Life is definitely digital. The genetic code of all living organisms are represented by a long sequence of simple molecules called nucleotides, or bases, which makes up the Deoxyribonucleic acid, better known as DNA. There are only four such nucleotides, and the entire genetic code of a human can be seen as a simple, though 3 billion long, string of the letters A, C, G, and T. Analyzing DNA data to gain increased biological understanding is much about searching in (long) strings for certain string patterns involving the letters A, C, G, and T. This is an integral part of *bioinformatics*, a scientific discipline addressing the use of computers to search for, explore, and use information about genes, nucleic acids, and proteins.

1 Basic Bioinformatics Examples in Python

Biopython



Biopython Tutorial and Cookbook

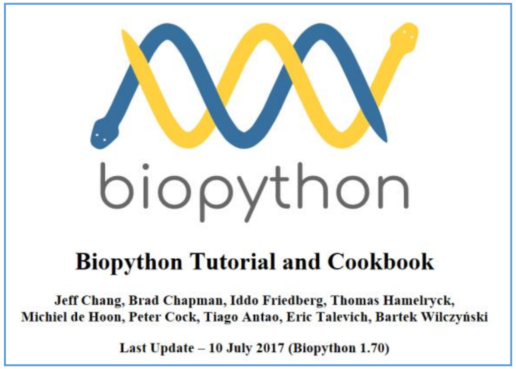
Jeff Chang, Brad Chapman, Iddo Friedberg, Thomas Hamelryck,
Michiel de Hoon, Peter Cock, Tiago Antao, Eric Talevich, Bartek Wilczyński

Last Update – 10 July 2017 (Biopython 1.70)

Biopython

Chapter 20 Cookbook – Cool things to do with it

- 20.1 Working with sequence files
  - 20.1.1 Filtering a sequence file
  - 20.1.2 Producing randomised genomes
  - 20.1.3 Translating a FASTA file of CDS entries
  - 20.1.4 Making the sequences in a FASTA file upper case
  - 20.1.5 Sorting a sequence file
  - 20.1.6 Simple quality filtering for FASTQ files
  - 20.1.7 Trimming off primer sequences
  - 20.1.8 Trimming off adaptor sequences
  - 20.1.9 Converting FASTQ files
  - 20.1.10 Converting FASTA and QUAL files into FASTQ files
  - 20.1.11 Indexing a FASTQ file
  - 20.1.12 Converting SFF files
  - 20.1.13 Identifying open reading frames
- 20.2 Sequence parsing plus simple plots
  - 20.2.1 Histogram of sequence lengths
  - 20.2.2 Plot of sequence GC%
  - 20.2.3 Nucleotide dot plots
  - 20.2.4 Plotting the quality scores of sequencing read data
- 20.3 Dealing with alignments
  - 20.3.1 Calculating summary information
  - 20.3.2 Calculating a quick consensus sequence
  - 20.3.3 Position Specific Score Matrices
  - 20.3.4 Information Content
- 20.4 Substitution Matrices
  - 20.4.1 Using common substitution matrices
  - 20.4.2 Creating your own substitution matrix from an alignment
- 20.5 BioSQL – storing sequences in a relational database



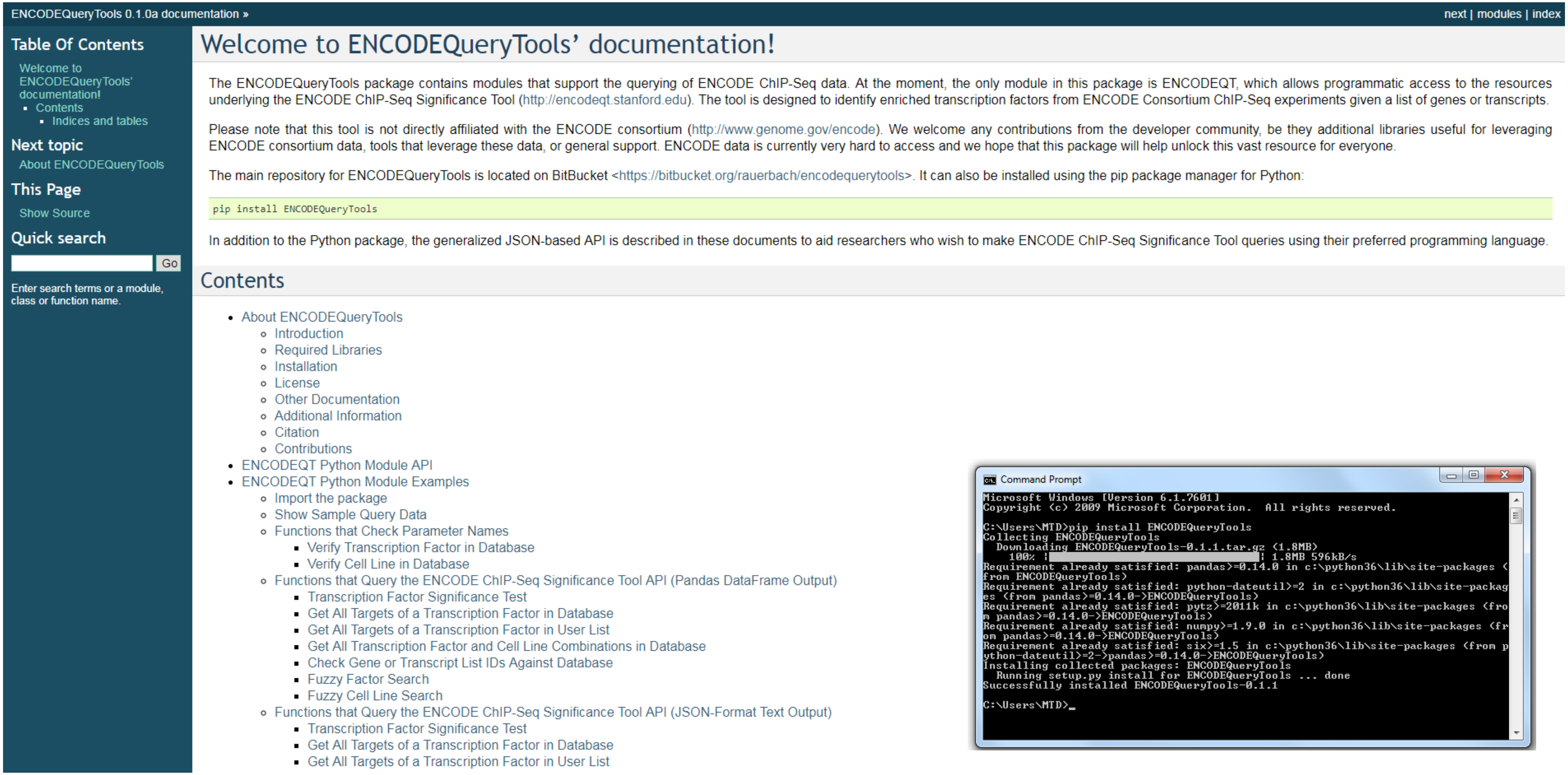
Biopython Tutorial and Cookbook

Jeff Chang, Brad Chapman, Iddo Friedberg, Thomas Hamelryck, Michiel de Hoon, Peter Cock, Tiago Antao, Eric Talevich, Bartek Wilczyński

Last Update – 10 July 2017 (Biopython 1.70)

Kingston University London

Python & Genomics



Enter search terms or a module, class or function name.

Welcome to ENCODEQueryTools' documentation!

The ENCODEQueryTools package contains modules that support the querying of ENCODE ChIP-Seq data. At the moment, the only module in this package is ENCODEQT, which allows programmatic access to the resources underlying the ENCODE ChIP-Seq Significance Tool (http://encodeqt.stanford.edu). The tool is designed to identify enriched transcription factors from ENCODE Consortium ChIP-Seq experiments given a list of genes or transcripts.

Please note that this tool is not directly affiliated with the ENCODE consortium (http://www.genome.gov/encode). We welcome any contributions from the developer community, be they additional libraries useful for leveraging ENCODE consortium data, tools that leverage these data, or general support. ENCODE data is currently very hard to access and we hope that this package will help unlock this vast resource for everyone.

The main repository for ENCODEQueryTools is located on BitBucket <https://bitbucket.org/rauerbach/encodequerytools>. It can also be installed using the pip package manager for Python:

```
pip install ENCODEQueryTools
```

In addition to the Python package, the generalized JSON-based API is described in these documents to aid researchers who wish to make ENCODE ChIP-Seq Significance Tool queries using their preferred programming language.

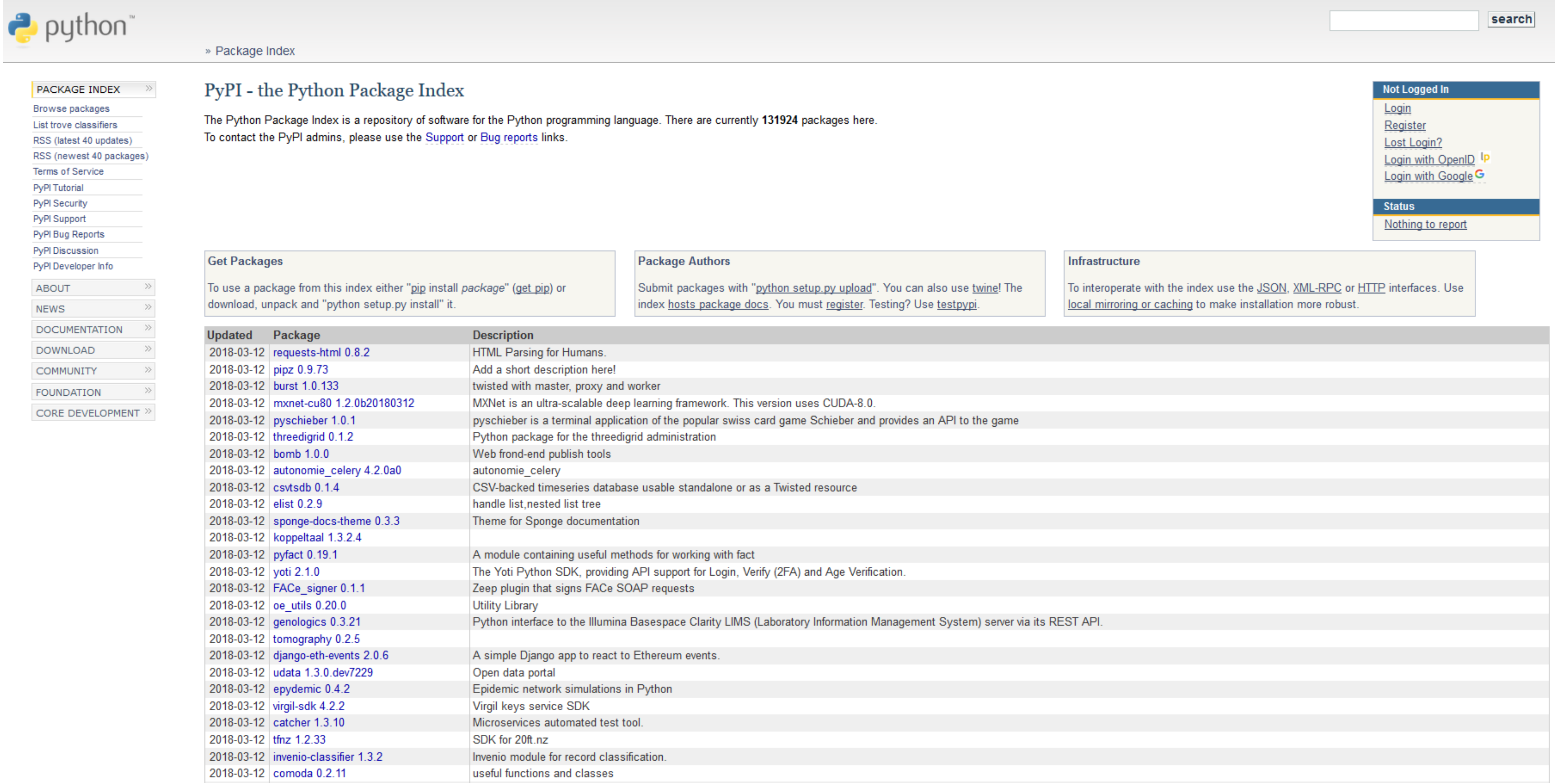
Contents

- About ENCODEQueryTools
  - Introduction
  - Required Libraries
  - Installation
  - License
  - Other Documentation
  - Additional Information
  - Citation
  - Contributions
- ENCODEQT Python Module API
- ENCODEQT Python Module Examples
  - Import the package
  - Show Sample Query Data
  - Functions that Check Parameter Names
    - Verify Transcription Factor in Database
    - Verify Cell Line in Database
  - Functions that Query the ENCODE ChIP-Seq Significance Tool API (Pandas DataFrame Output)
    - Transcription Factor Significance Test
    - Get All Targets of a Transcription Factor in Database
    - Get All Targets of a Transcription Factor in User List
    - Get All Transcription Factor and Cell Line Combinations in Database
    - Check Gene or Transcript List IDs Against Database
    - Fuzzy Factor Search
    - Fuzzy Cell Line Search
  - Functions that Query the ENCODE ChIP-Seq Significance Tool API (JSON-Format Text Output)
    - Transcription Factor Significance Test
    - Get All Targets of a Transcription Factor in Database
    - Get All Targets of a Transcription Factor in User List

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\MTD>pip install ENCODEQueryTools
Collecting ENCODEQueryTools
  Downloading ENCODEQueryTools-0.1.1.tar.gz (1.8MB)
    100% |████████████████████████████████| 1.8MB 596kB/s
Requirement already satisfied: pandas>=0.14.0 in c:\python36\lib\site-packages (
from ENCODEQueryTools)
Requirement already satisfied: python-dateutil>=2 in c:\python36\lib\site-packag
es (from pandas>=0.14.0->ENCODEQueryTools)
Requirement already satisfied: pytz>=2011k in c:\python36\lib\site-packages (fro
m pandas>=0.14.0->ENCODEQueryTools)
Requirement already satisfied: numpy>=1.9.0 in c:\python36\lib\site-packages (fr
om pandas>=0.14.0->ENCODEQueryTools)
Requirement already satisfied: six>=1.5 in c:\python36\lib\site-packages (from p
ython-dateutil>=2->pandas>=0.14.0->ENCODEQueryTools)
Installing collected packages: ENCODEQueryTools
  Running setup.py install for ENCODEQueryTools ... done
Successfully installed ENCODEQueryTools-0.1.1

C:\Users\MTD>_
```

Kingston University London

Python Packages



PyPI - the Python Package Index

The Python Package Index is a repository of software for the Python programming language. There are currently **131924** packages here.
To contact the PyPI admins, please use the Support or Bug reports links.

Get Packages

To use a package from this index either "pip install *package*" (get pip) or download, unpack and "python setup.py install" it.

Package Authors

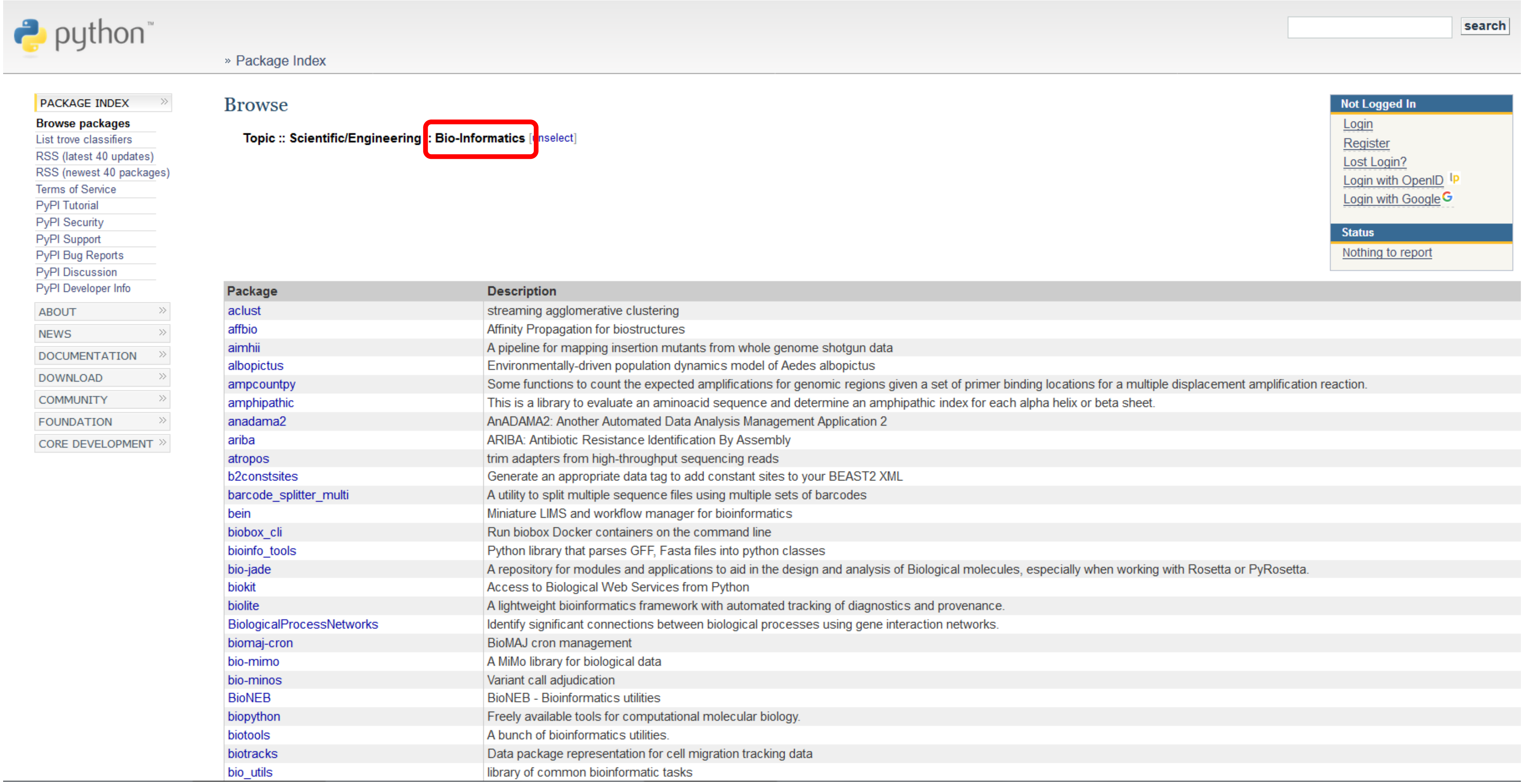
Submit packages with "python setup.py upload". You can also use twine! The index hosts package docs. You must register. Testing? Use testpypi.

Infrastructure

To interoperate with the index use the JSON, XML-RPC or HTTP interfaces. Use local mirroring or caching to make installation more robust.

Updated	Package	Description
2018-03-12	requests-html 0.8.2	HTML Parsing for Humans.
2018-03-12	pipz 0.9.73	Add a short description here!
2018-03-12	burst 1.0.133	twisted with master, proxy and worker
2018-03-12	mxnet-cu80 1.2.0b20180312	MXNet is an ultra-scalable deep learning framework. This version uses CUDA-8.0.
2018-03-12	pyschieber 1.0.1	pyschieber is a terminal application of the popular swiss card game Schieber and provides an API to the game
2018-03-12	threedigrid 0.1.2	Python package for the threedigrid administration
2018-03-12	bomb 1.0.0	Web frond-end publish tools
2018-03-12	autonomie_celery 4.2.0a0	autonomie_celery
2018-03-12	csvtsdb 0.1.4	CSV-backed timeseries database usable standalone or as a Twisted resource
2018-03-12	elist 0.2.9	handle list,nested list tree
2018-03-12	sponge-docs-theme 0.3.3	Theme for Sponge documentation
2018-03-12	koppeltaal 1.3.2.4	
2018-03-12	pyfact 0.19.1	A module containing useful methods for working with fact
2018-03-12	yoti 2.1.0	The Yoti Python SDK, providing API support for Login, Verify (2FA) and Age Verification.
2018-03-12	FACe_signer 0.1.1	Zeep plugin that signs FACe SOAP requests
2018-03-12	oe_utils 0.20.0	Utility Library
2018-03-12	genologics 0.3.21	Python interface to the Illumina Basespace Clarity LIMS (Laboratory Information Management System) server via its REST API.
2018-03-12	tomography 0.2.5	
2018-03-12	django-eth-events 2.0.6	A simple Django app to react to Ethereum events.
2018-03-12	udata 1.3.0.dev7229	Open data portal
2018-03-12	epydemic 0.4.2	Epidemic network simulations in Python
2018-03-12	virgil-sdk 4.2.2	Virgil keys service SDK
2018-03-12	catcher 1.3.10	Microservices automated test tool.
2018-03-12	tfnz 1.2.33	SDK for 20ft.nz
2018-03-12	invenio-classifier 1.3.2	Invenio module for record classification.
2018-03-12	comoda 0.2.11	useful functions and classes

Python Packages

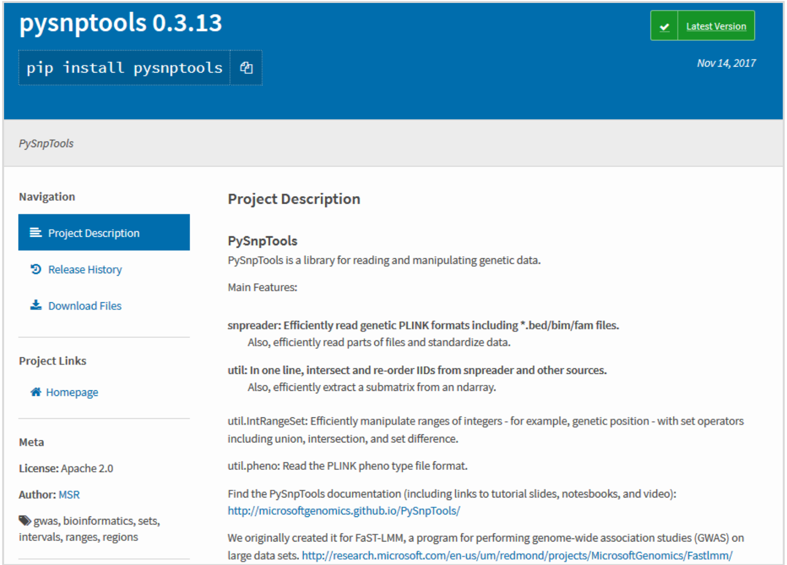


Browse

Topic :: Scientific/Engineering :: Bio-Informatics [unselect]

Package	Description
aclust	streaming agglomerative clustering
affbio	Affinity Propagation for biostructures
aimhii	A pipeline for mapping insertion mutants from whole genome shotgun data
albopictus	Environmentally-driven population dynamics model of Aedes albopictus
ampcountpy	Some functions to count the expected amplifications for genomic regions given a set of primer binding locations for a multiple displacement amplification reaction.
amphipathic	This is a library to evaluate an aminoacid sequence and determine an amphipathic index for each alpha helix or beta sheet.
anadama2	AnADAMA2: Another Automated Data Analysis Management Application 2
ariba	ARIBA: Antibiotic Resistance Identification By Assembly
atropos	trim adapters from high-throughput sequencing reads
b2constsites	Generate an appropriate data tag to add constant sites to your BEAST2 XML
barcode_splitter_multi	A utility to split multiple sequence files using multiple sets of barcodes
bein	Miniature LIMS and workflow manager for bioinformatics
biobox_cli	Run biobox Docker containers on the command line
bioinfo_tools	Python library that parses GFF, Fasta files into python classes
bio-jade	A repository for modules and applications to aid in the design and analysis of Biological molecules, especially when working with Rosetta or PyRosetta.
biokit	Access to Biological Web Services from Python
biolite	A lightweight bioinformatics framework with automated tracking of diagnostics and provenance.
BiologicalProcessNetworks	Identify significant connections between biological processes using gene interaction networks.
biomaj-cron	BioMAJ cron management
bio-mimo	A MiMo library for biological data
bio-minos	Variant call adjudication
BioNEB	BioNEB - Bioinformatics utilities
biopython	Freely available tools for computational molecular biology.
biotools	A bunch of bioinformatics utilities.
biotracks	Data package representation for cell migration tracking data
bio_utils	library of common bioinformatic tasks

Python Packages



pysnptools 0.3.13

✓ Latest Version

```
pip install pysnptools
```

Nov 14, 2017

PySnpTools

Navigation

- Project Description
- Release History
- Download Files

Project Links

- Homepage

Meta

License: Apache 2.0

Author: MSR

gwas, bioinformatics, sets, intervals, ranges, regions

Project Description

PySnpTools

PySnpTools is a library for reading and manipulating genetic data.

Main Features:

snpreader: Efficiently read genetic PLINK formats including *.bed/bim/fam files.
Also, efficiently read parts of files and standardize data.

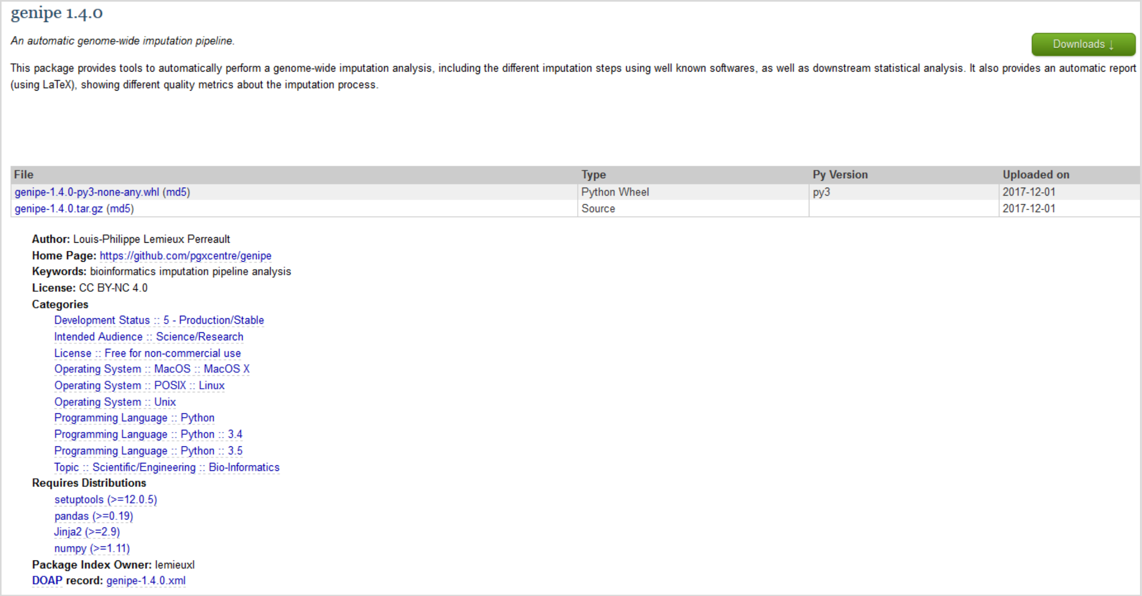
util: In one line, intersect and re-order IIDs from snpreader and other sources.
Also, efficiently extract a submatrix from an ndarray.

util.IntRangeSet: Efficiently manipulate ranges of integers - for example, genetic position - with set operators including union, intersection, and set difference.

util.pheno: Read the PLINK pheno type file format.

Find the PySnpTools documentation (including links to tutorial slides, notesbooks, and video): http://microsoftgenomics.github.io/PySnpTools/

We originally created it for FaST-LMM, a program for performing genome-wide association studies (GWAS) on large data sets. http://research.microsoft.com/en-us/um/redmond/projects/MicrosoftGenomics/Fastlmm/



genipe 1.4.0

An automatic genome-wide imputation pipeline.

Downloads ↓

This package provides tools to automatically perform a genome-wide imputation analysis, including the different imputation steps using well known softwares, as well as downstream statistical analysis. It also provides an automatic report (using LaTeX), showing different quality metrics about the imputation process.

File	Type	Py Version	Uploaded on
genipe-1.4.0-py3-none-any.whl (md5)	Python Wheel	py3	2017-12-01
genipe-1.4.0.tar.gz (md5)	Source		2017-12-01

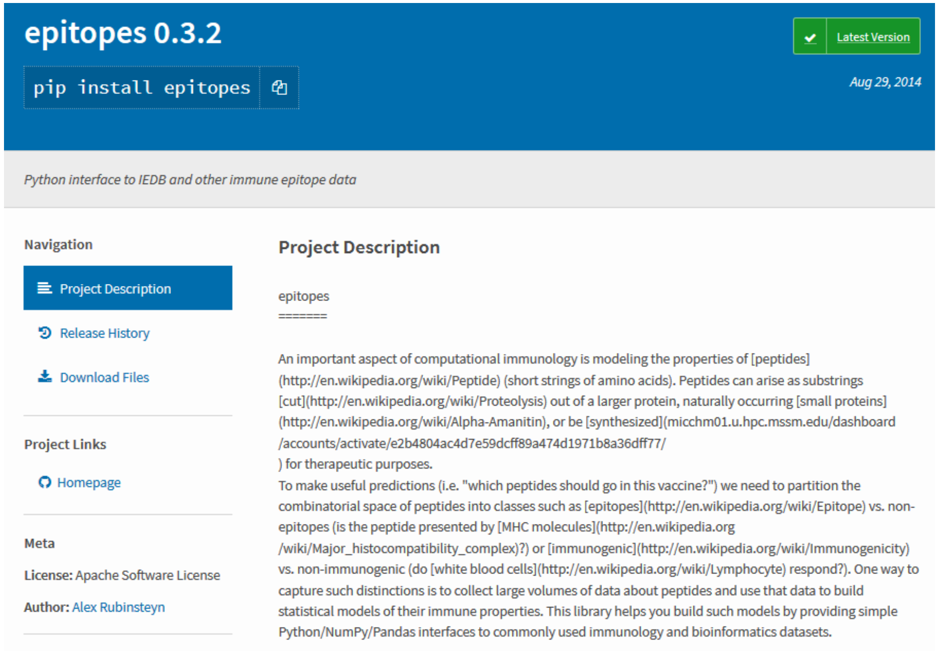
Author: Louis-Philippe Lemieux Perreault
Home Page: https://github.com/pgxcentre/genipe
Keywords: bioinformatics imputation pipeline analysis
License: CC BY-NC 4.0
Categories

- Development Status :: 5 - Production/Stable
- Intended Audience :: Science/Research
- License :: Free for non-commercial use
- Operating System :: MacOS :: MacOS X
- Operating System :: POSIX :: Linux
- Operating System :: Unix
- Programming Language :: Python
- Programming Language :: Python :: 3.4
- Programming Language :: Python :: 3.5
- Topic :: Scientific/Engineering :: Bio-Informatics

Requires Distributions

- setuptools (>=12.0.5)
- pandas (>=0.19)
- Jinja2 (>=2.9)
- numpy (>=1.11)

Package Index Owner: lemieuxl
DOAP record: genipe-1.4.0.xml



epitopes 0.3.2

✓ Latest Version

```
pip install epitopes
```

Aug 29, 2014

Python interface to IEDB and other immune epitope data

Navigation

- Project Description
- Release History
- Download Files

Project Links

- Homepage

Meta

License: Apache Software License

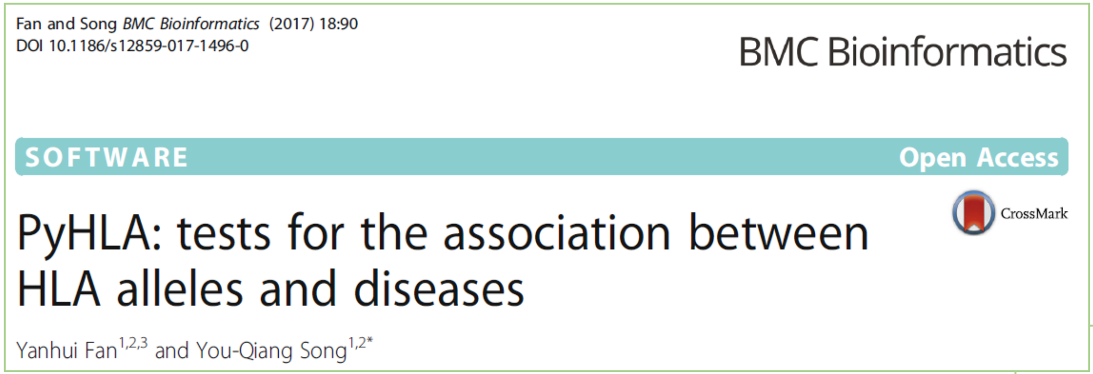
Author: Alex Rubinsteyn

Project Description

epitopes
=======

An important aspect of computational immunology is modeling the properties of [peptides](http://en.wikipedia.org/wiki/Peptide) (short strings of amino acids). Peptides can arise as substrings [cut](http://en.wikipedia.org/wiki/Proteolysis) out of a larger protein, naturally occurring [small proteins](http://en.wikipedia.org/wiki/Alpha-Amanitin), or be [synthesized](micchm01.u.hpc.mssm.edu/dashboard/accounts/activate/e2b4804ac4d7e59dcff89a474d1971b8a36dff77/
) for therapeutic purposes.
To make useful predictions (i.e. "which peptides should go in this vaccine?") we need to partition the combinatorial space of peptides into classes such as [epitopes](http://en.wikipedia.org/wiki/Epitope) vs. non-epitopes (is the peptide presented by [MHC molecules](http://en.wikipedia.org/wiki/Major_histocompatibility_complex)?) or [immunogenic](http://en.wikipedia.org/wiki/Immunogenicity) vs. non-immunogenic (do [white blood cells](http://en.wikipedia.org/wiki/Lymphocyte) respond?). One way to capture such distinctions is to collect large volumes of data about peptides and use that data to build statistical models of their immune properties. This library helps you build such models by providing simple Python/NumPy/Pandas interfaces to commonly used immunology and bioinformatics datasets.

Python Packages



Fan and Song *BMC Bioinformatics* (2017) 18:90
DOI 10.1186/s12859-017-1496-0

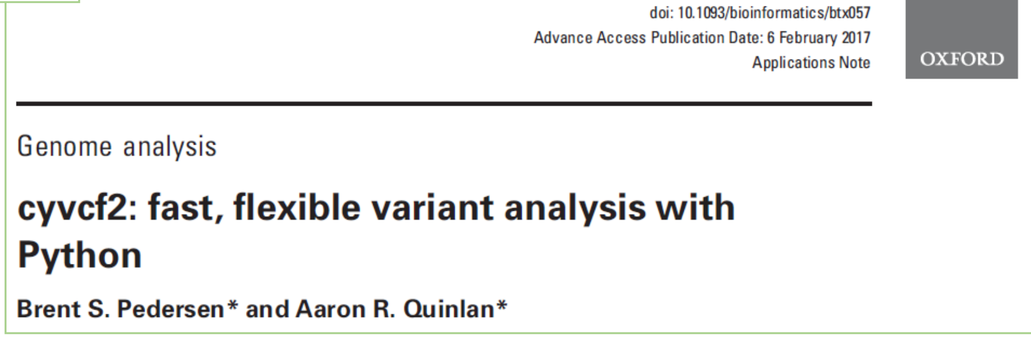
BMC Bioinformatics

SOFTWARE Open Access

CrossMark

PyHLA: tests for the association between HLA alleles and diseases

Yanhui Fan[1,2,3] and You-Qiang Song[1,2*]



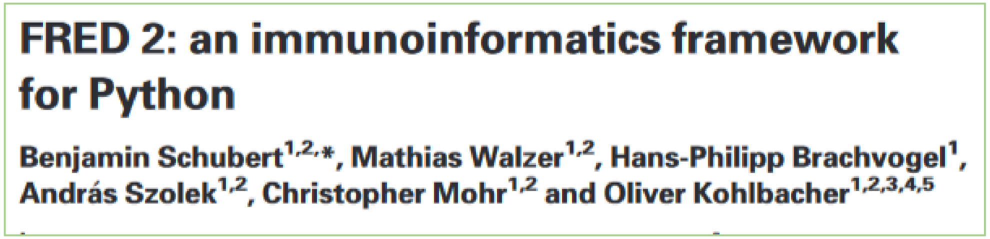
Bioinformatics, 33(12), 2017, 1867–1869
doi: 10.1093/bioinformatics/btx057
Advance Access Publication Date: 6 February 2017
Applications Note

OXFORD

Genome analysis

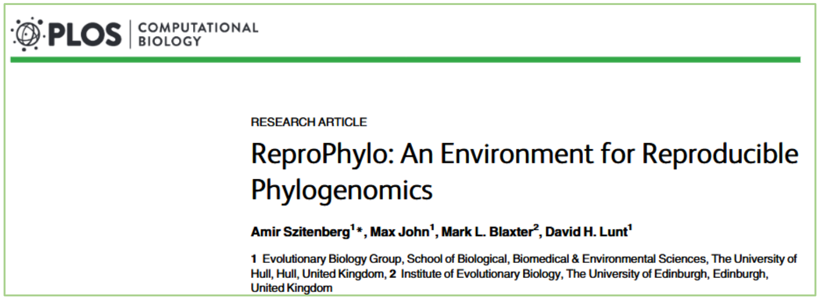
cyvcf2: fast, flexible variant analysis with Python

Brent S. Pedersen* and Aaron R. Quinlan*



FRED 2: an immunoinformatics framework for Python

Benjamin Schubert[1,2,*], Mathias Walzer[1,2], Hans-Philipp Brachvogel[1], András Szolek[1,2], Christopher Mohr[1,2] and Oliver Kohlbacher[1,2,3,4,5]



PLOS | COMPUTATIONAL BIOLOGY

RESEARCH ARTICLE

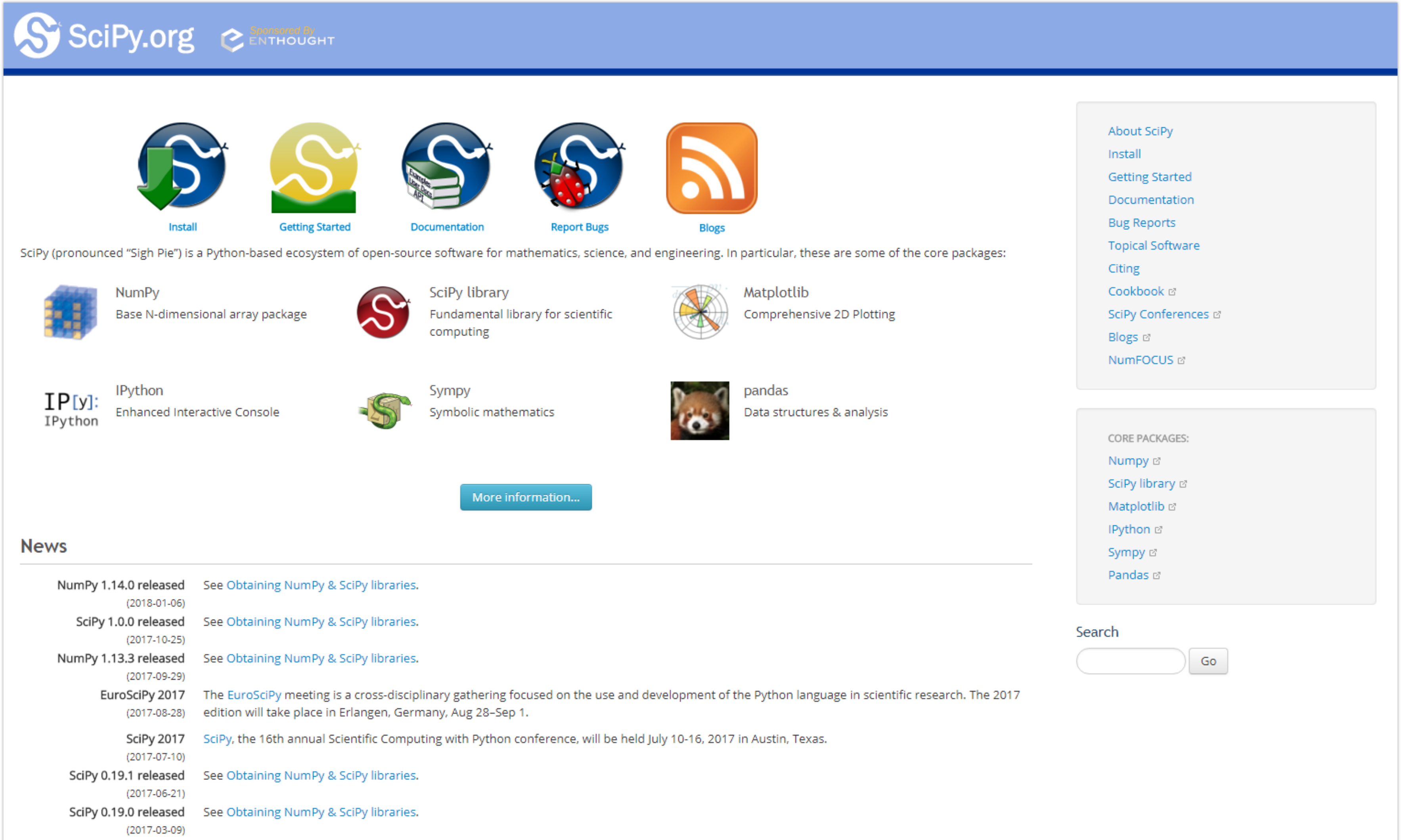
ReproPhylo: An Environment for Reproducible Phylogenomics

Amir Szitenberg[1]*, Max John[1], Mark L. Blaxter[2], David H. Lunt[1]

1 Evolutionary Biology Group, School of Biological, Biomedical & Environmental Sciences, The University of Hull, Hull, United Kingdom, 2 Institute of Evolutionary Biology, The University of Edinburgh, Edinburgh, United Kingdom

Kingston University London

Python Packages



SciPy (pronounced "Sigh Pie") is a Python-based ecosystem of open-source software for mathematics, science, and engineering. In particular, these are some of the core packages:

NumPy
Base N-dimensional array package

SciPy library
Fundamental library for scientific computing

Matplotlib
Comprehensive 2D Plotting

IPython
Enhanced Interactive Console

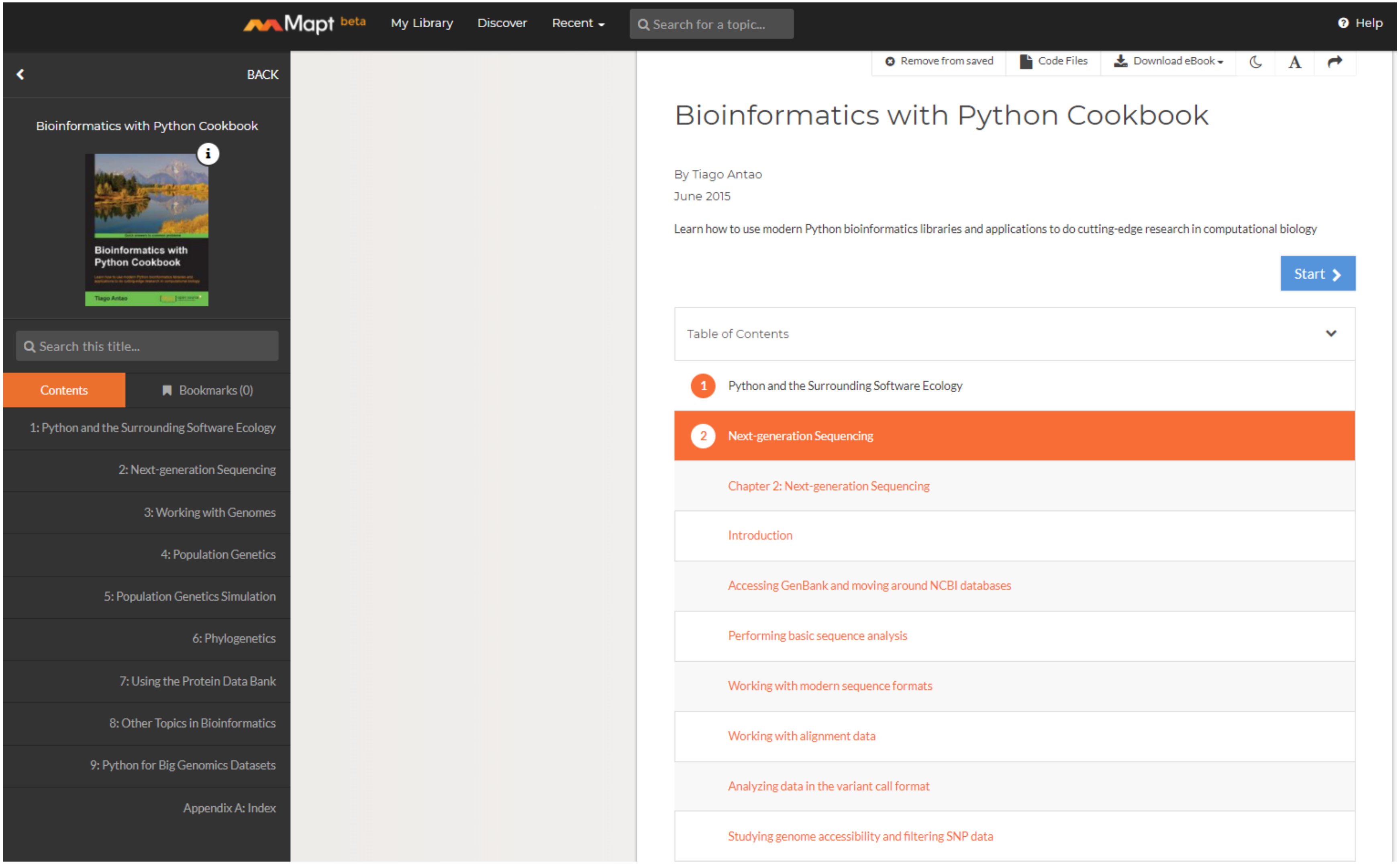
Sympy
Symbolic mathematics

pandas
Data structures & analysis

News

NumPy 1.14.0 released (2018-01-06)	See Obtaining NumPy & SciPy libraries.
SciPy 1.0.0 released (2017-10-25)	See Obtaining NumPy & SciPy libraries.
NumPy 1.13.3 released (2017-09-29)	See Obtaining NumPy & SciPy libraries.
EuroSciPy 2017 (2017-08-28)	The EuroSciPy meeting is a cross-disciplinary gathering focused on the use and development of the Python language in scientific research. The 2017 edition will take place in Erlangen, Germany, Aug 28–Sep 1.
SciPy 2017 (2017-07-10)	SciPy, the 16th annual Scientific Computing with Python conference, will be held July 10-16, 2017 in Austin, Texas.
SciPy 0.19.1 released (2017-06-21)	See Obtaining NumPy & SciPy libraries.
SciPy 0.19.0 released (2017-03-09)	See Obtaining NumPy & SciPy libraries.

Book: Python for Bioinformatics



Bioinformatics with Python Cookbook

1: Python and the Surrounding Software Ecology
2: Next-generation Sequencing
3: Working with Genomes
4: Population Genetics
5: Population Genetics Simulation
6: Phylogenetics
7: Using the Protein Data Bank
8: Other Topics in Bioinformatics
9: Python for Big Genomics Datasets
Appendix A: Index

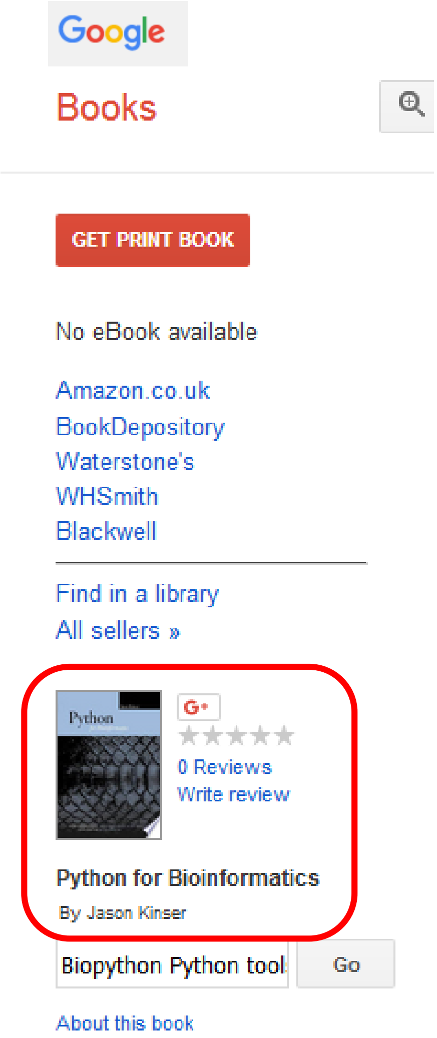
Bioinformatics with Python Cookbook

By Tiago Antao
June 2015

Learn how to use modern Python bioinformatics libraries and applications to do cutting-edge research in computational biology

1 Python and the Surrounding Software Ecology
2 Next-generation Sequencing
Chapter 2: Next-generation Sequencing
Introduction
Accessing GenBank and moving around NCBI databases
Performing basic sequence analysis
Working with modern sequence formats
Working with alignment data
Analyzing data in the variant call format
Studying genome accessibility and filtering SNP data

Book: Python for Bioinformatics



Google

Books

GET PRINT BOOK

No eBook available

Amazon.co.uk
BookDepository
Waterstone's
WHSmith
Blackwell

Find in a library
All sellers »

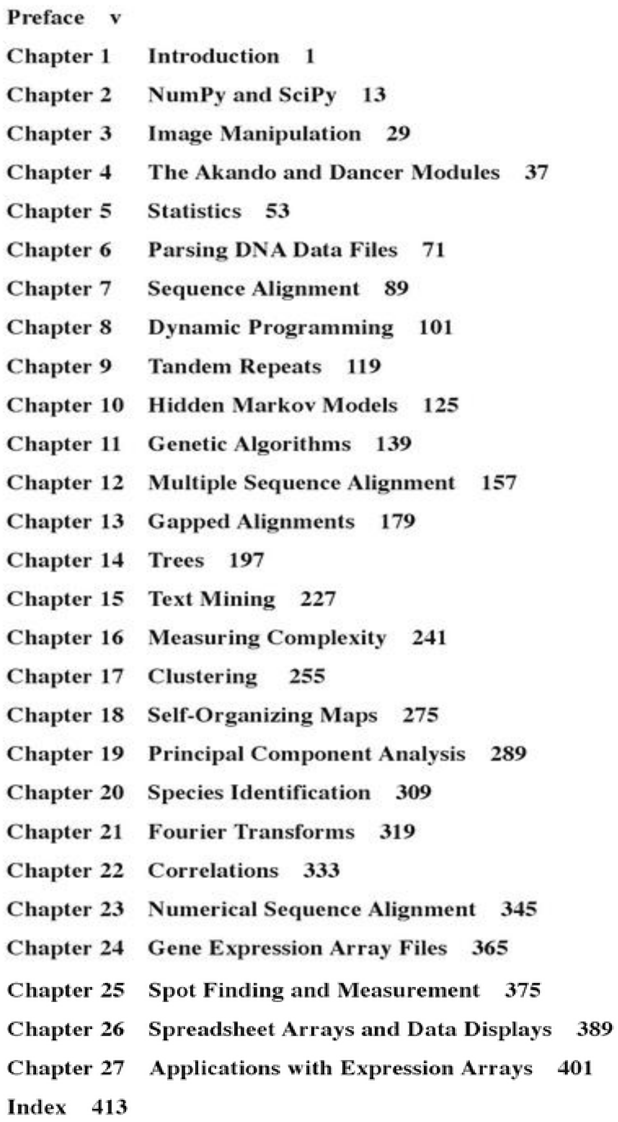
0 Reviews
Write review

Python for Bioinformatics
By Jason Kinser

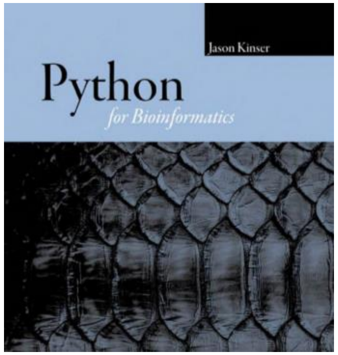
Biopython Python tool

Go

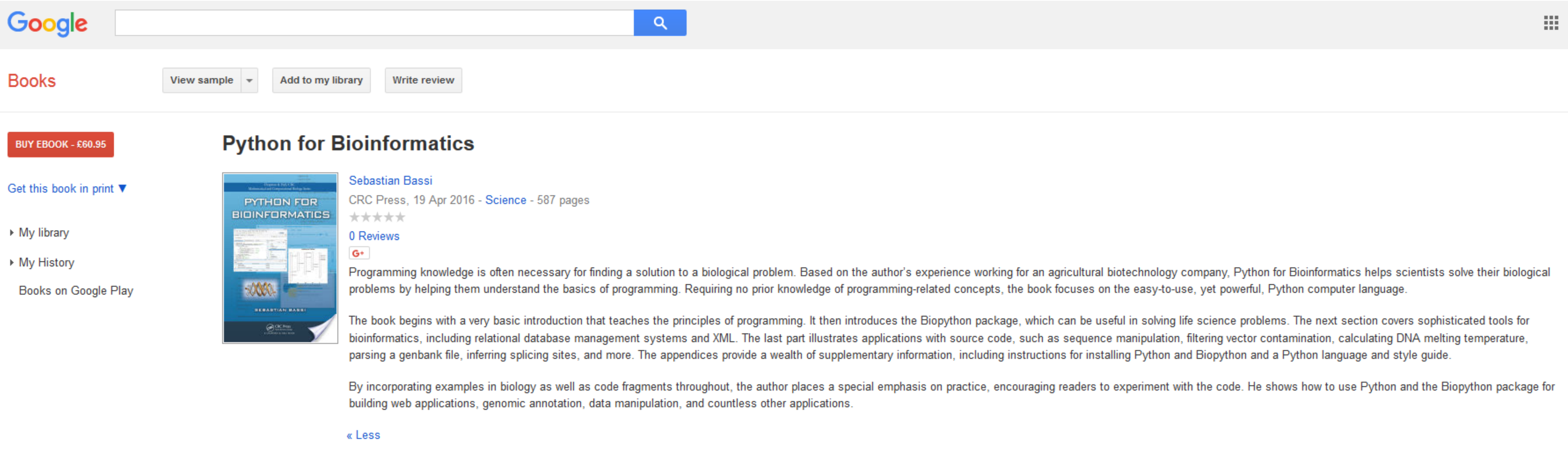
About this book



Preface		v
Chapter 1	Introduction	1
Chapter 2	NumPy and SciPy	13
Chapter 3	Image Manipulation	29
Chapter 4	The Akando and Dancer Modules	37
Chapter 5	Statistics	53
Chapter 6	Parsing DNA Data Files	71
Chapter 7	Sequence Alignment	89
Chapter 8	Dynamic Programming	101
Chapter 9	Tandem Repeats	119
Chapter 10	Hidden Markov Models	125
Chapter 11	Genetic Algorithms	139
Chapter 12	Multiple Sequence Alignment	157
Chapter 13	Gapped Alignments	179
Chapter 14	Trees	197
Chapter 15	Text Mining	227
Chapter 16	Measuring Complexity	241
Chapter 17	Clustering	255
Chapter 18	Self-Organizing Maps	275
Chapter 19	Principal Component Analysis	289
Chapter 20	Species Identification	309
Chapter 21	Fourier Transforms	319
Chapter 22	Correlations	333
Chapter 23	Numerical Sequence Alignment	345
Chapter 24	Gene Expression Array Files	365
Chapter 25	Spot Finding and Measurement	375
Chapter 26	Spreadsheet Arrays and Data Displays	389
Chapter 27	Applications with Expression Arrays	401
Index		413



Book: Python for Bioinformatics



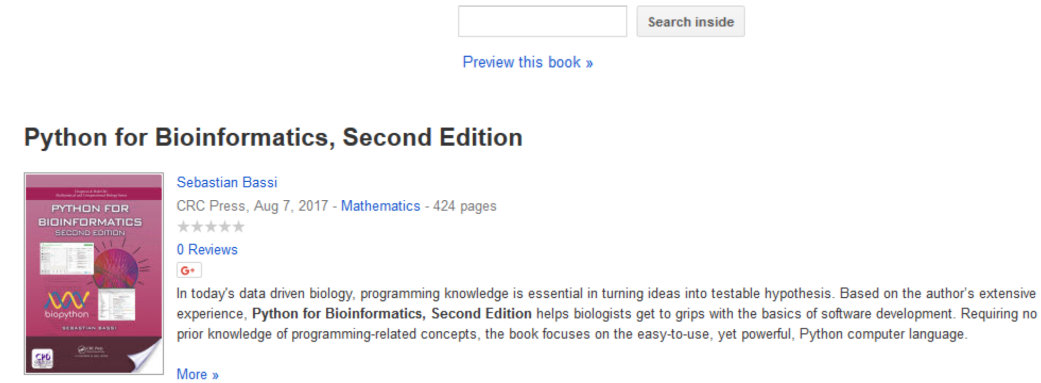
Python for Bioinformatics

Sebastian Bassi
CRC Press, 19 Apr 2016 - Science - 587 pages
☆☆☆☆☆
0 Reviews

Programming knowledge is often necessary for finding a solution to a biological problem. Based on the author's experience working for an agricultural biotechnology company, Python for Bioinformatics helps scientists solve their biological problems by helping them understand the basics of programming. Requiring no prior knowledge of programming-related concepts, the book focuses on the easy-to-use, yet powerful, Python computer language.

The book begins with a very basic introduction that teaches the principles of programming. It then introduces the Biopython package, which can be useful in solving life science problems. The next section covers sophisticated tools for bioinformatics, including relational database management systems and XML. The last part illustrates applications with source code, such as sequence manipulation, filtering vector contamination, calculating DNA melting temperature, parsing a genbank file, inferring splicing sites, and more. The appendices provide a wealth of supplementary information, including instructions for installing Python and Biopython and a Python language and style guide.

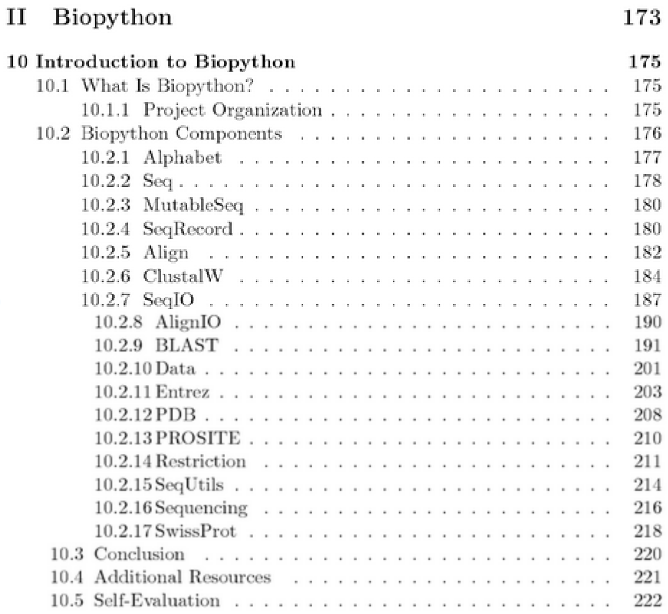
By incorporating examples in biology as well as code fragments throughout, the author places a special emphasis on practice, encouraging readers to experiment with the code. He shows how to use Python and the Biopython package for building web applications, genomic annotation, data manipulation, and countless other applications.



Python for Bioinformatics, Second Edition

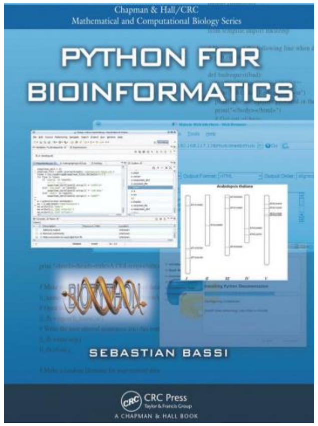
Sebastian Bassi
CRC Press, Aug 7, 2017 - Mathematics - 424 pages
☆☆☆☆☆
0 Reviews

In today's data driven biology, programming knowledge is essential in turning ideas into testable hypothesis. Based on the author's extensive experience, **Python for Bioinformatics, Second Edition** helps biologists get to grips with the basics of software development. Requiring no prior knowledge of programming-related concepts, the book focuses on the easy-to-use, yet powerful, Python computer language.



II Biopython 173

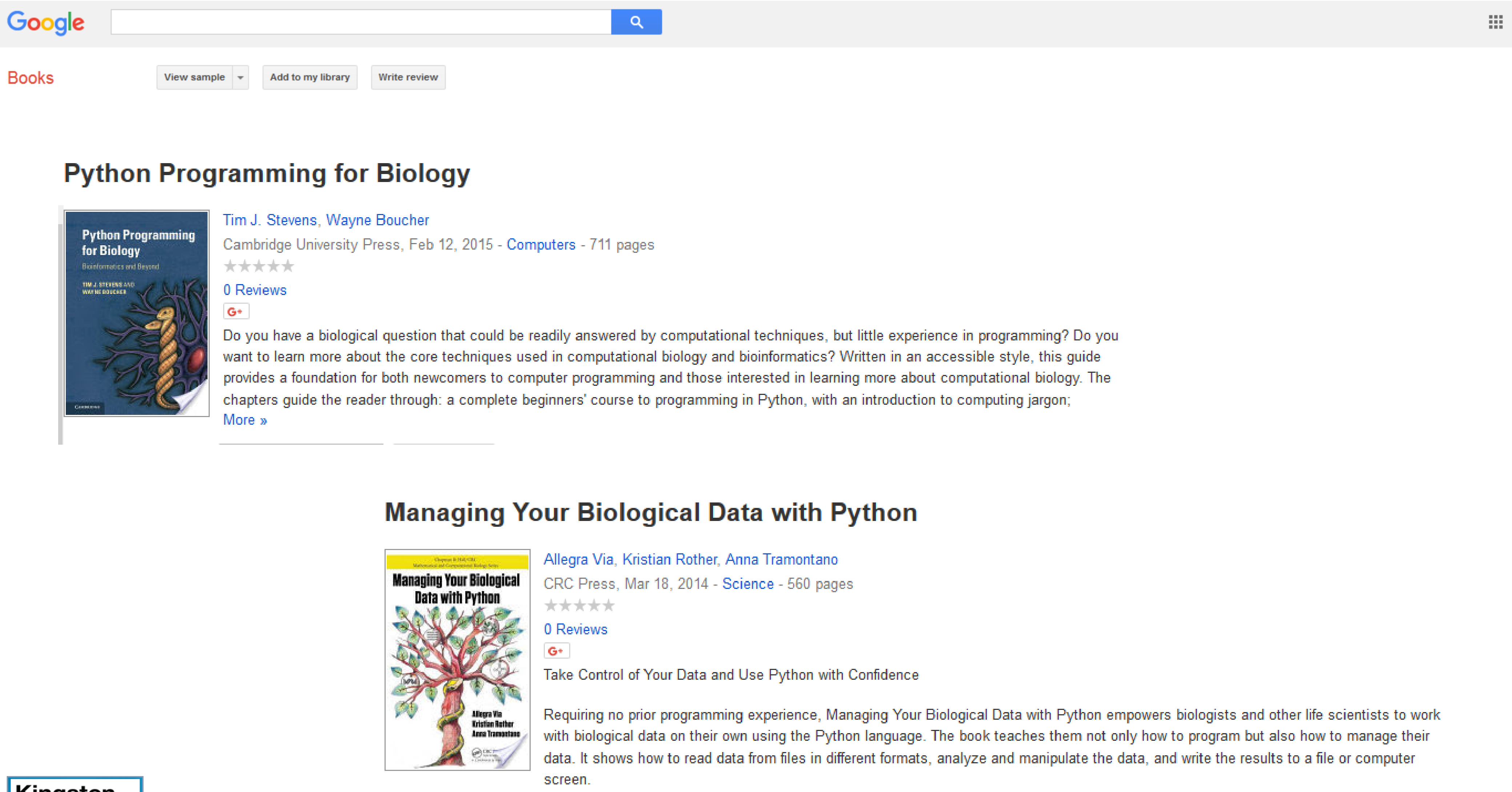
10 Introduction to Biopython 175
10.1 What Is Biopython? 175
10.1.1 Project Organization 175
10.2 Biopython Components 176
10.2.1 Alphabet 177
10.2.2 Seq 178
10.2.3 MutableSeq 180
10.2.4 SeqRecord 180
10.2.5 Align 182
10.2.6 ClustalW 184
10.2.7 SeqIO 187
10.2.8 AlignIO 190
10.2.9 BLAST 191
10.2.10 Data 201
10.2.11 Entrez 203
10.2.12 PDB 208
10.2.13 PROSITE 210
10.2.14 Restriction 211
10.2.15 SeqUtils 214
10.2.16 Sequencing 216
10.2.17 SwissProt 218
10.3 Conclusion 220
10.4 Additional Resources 221
10.5 Self-Evaluation 222



Chapman & Hall/CRC
Mathematical and Computational Biology Series
PYTHON FOR BIOINFORMATICS
SEBASTIAN BASSI
CRC Press

Kingston University London

Book: Python for Biologists


Google
Books
View sample
Add to my library
Write review
Python Programming for Biology
Bioinformatics and Beyond
TIM J. STEVENS AND WAYNE BOUCHER
Managing Your Biological Data with Python
Allegra Via
Kristian Rother
Anna Tramontano

Python Programming for Biology

Tim J. Stevens, Wayne Boucher

Cambridge University Press, Feb 12, 2015 - Computers - 711 pages

★★★★★

0 Reviews

Do you have a biological question that could be readily answered by computational techniques, but little experience in programming? Do you want to learn more about the core techniques used in computational biology and bioinformatics? Written in an accessible style, this guide provides a foundation for both newcomers to computer programming and those interested in learning more about computational biology. The chapters guide the reader through: a complete beginners' course to programming in Python, with an introduction to computing jargon;

More »

Managing Your Biological Data with Python

Allegra Via, Kristian Rother, Anna Tramontano

CRC Press, Mar 18, 2014 - Science - 560 pages

★★★★★

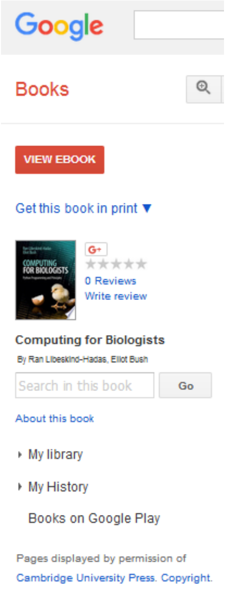
0 Reviews

Take Control of Your Data and Use Python with Confidence

Requiring no prior programming experience, Managing Your Biological Data with Python empowers biologists and other life scientists to work with biological data on their own using the Python language. The book teaches them not only how to program but also how to manage their data. It shows how to read data from files in different formats, analyze and manipulate the data, and write the results to a file or computer screen.

Kingston University London

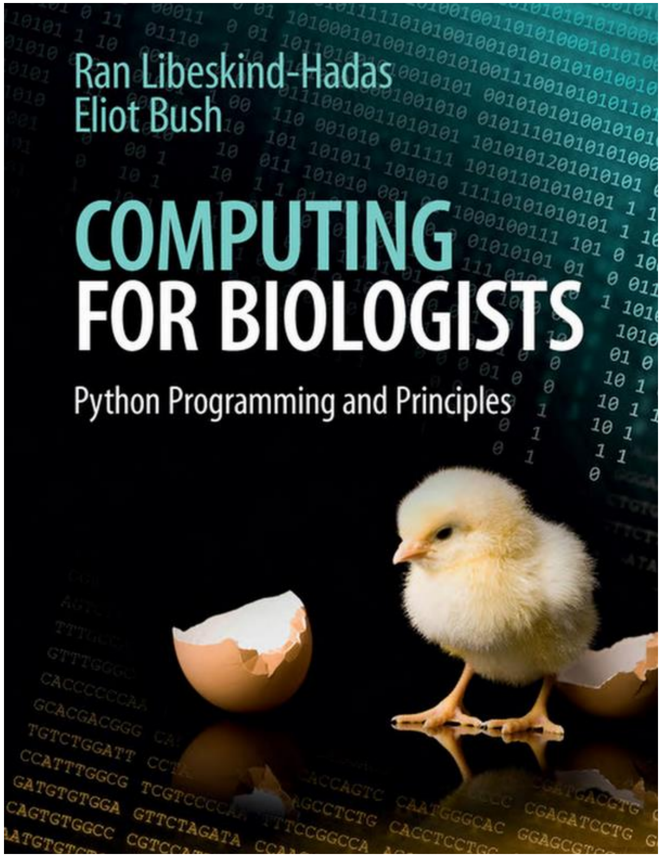
Book: Python for Biologists

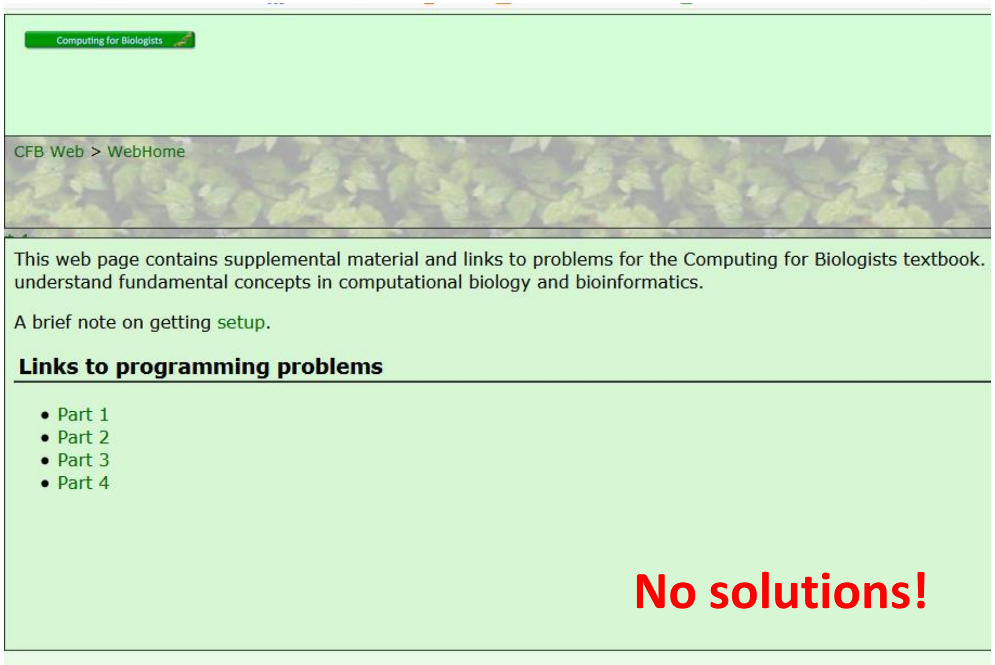


Computing for Biologists

By Ran Libeskind-Hadas, Eliot Bush

Pages displayed by permission of Cambridge University Press. Copyright.





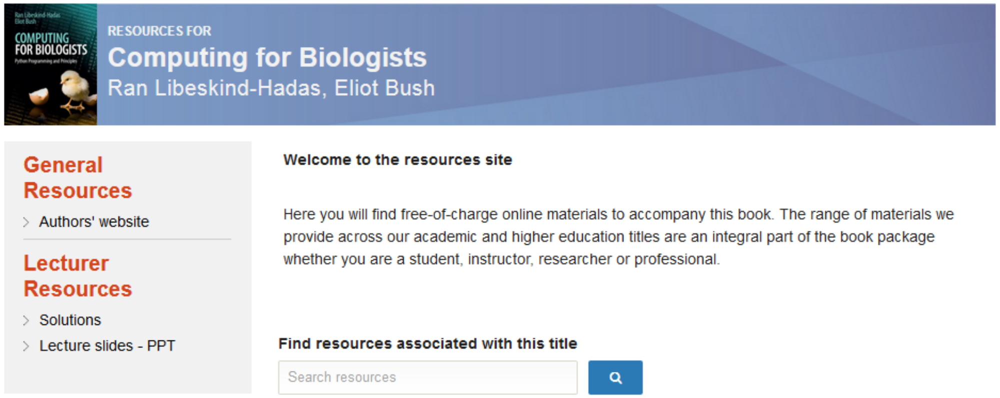
This web page contains supplemental material and links to problems for the Computing for Biologists textbook.
understand fundamental concepts in computational biology and bioinformatics.

A brief note on getting setup.

Links to programming problems

- Part 1
- Part 2
- Part 3
- Part 4

No solutions!



RESOURCES FOR

Computing for Biologists

Ran Libeskind-Hadas, Eliot Bush

General Resources

- Authors' website

Lecturer Resources

- Solutions
- Lecture slides - PPT

Welcome to the resources site

Here you will find free-of-charge online materials to accompany this book. The range of materials we provide across our academic and higher education titles are an integral part of the book package whether you are a student, instructor, researcher or professional.

Find resources associated with this title

Kingston University London

Book: Python for Biologists

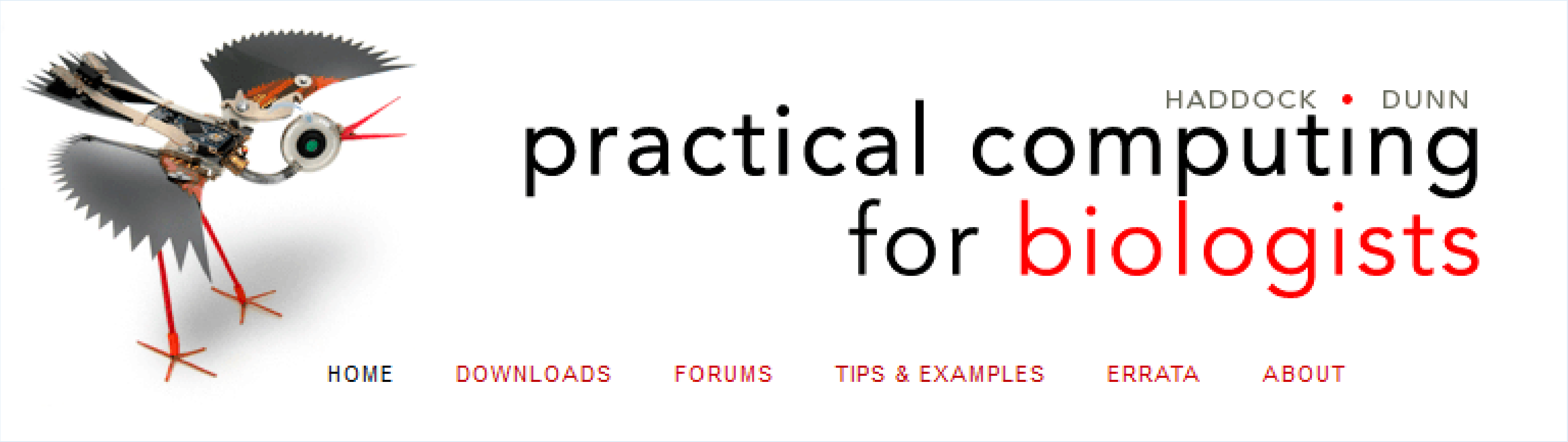
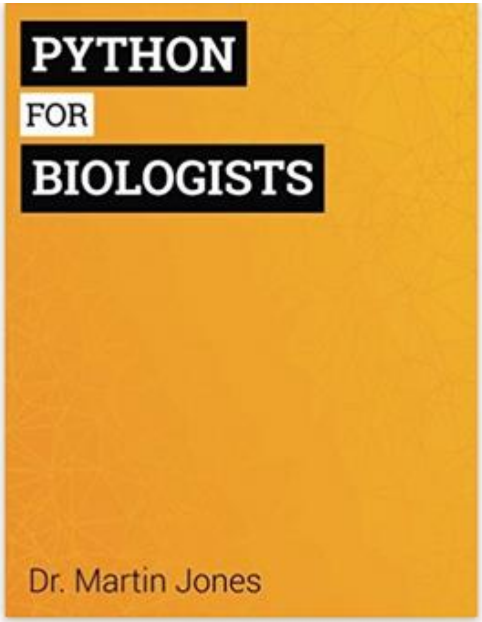


Table of Contents

Quick reference

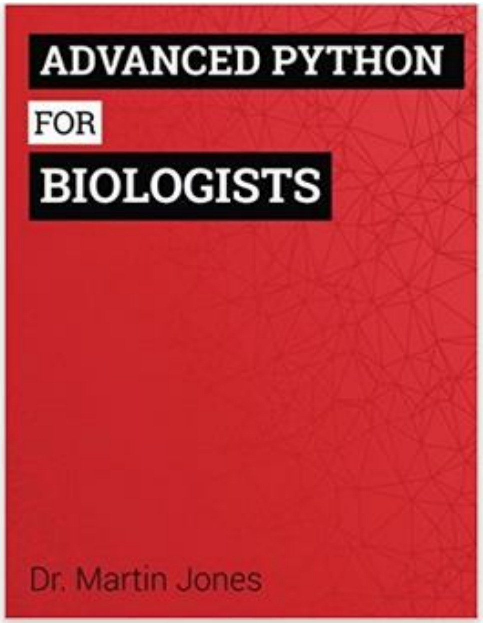
Examples

The Only Books You Need



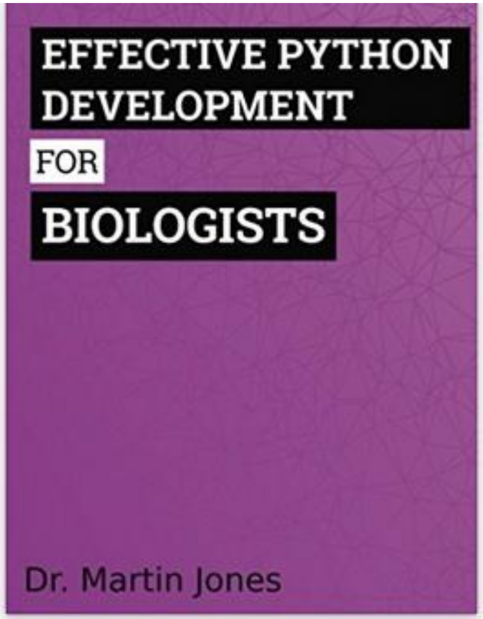
PYTHON
FOR
BIOLOGISTS

Dr. Martin Jones



ADVANCED PYTHON
FOR
BIOLOGISTS

Dr. Martin Jones

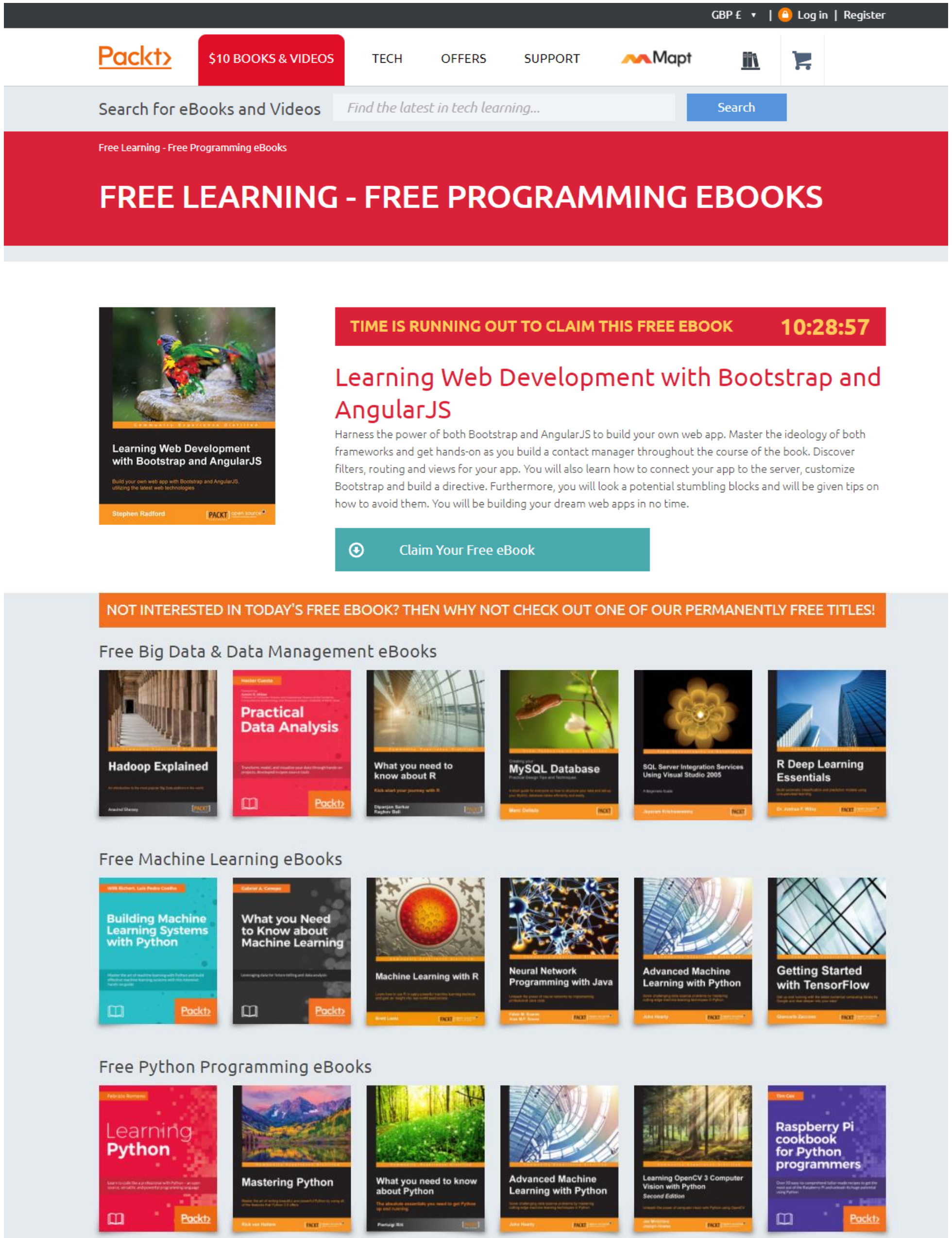


EFFECTIVE PYTHON DEVELOPMENT
FOR
BIOLOGISTS

Dr. Martin Jones

See also: https://pythonforbiologists.com

General Python Books



GBP £ ▾ | Log in | Register
Packt›
$10 BOOKS & VIDEOS
TECH
OFFERS
SUPPORT
Mapt

Search for eBooks and Videos

Find the latest in tech learning...
Search

Free Learning - Free Programming eBooks

FREE LEARNING - FREE PROGRAMMING EBOOKS

TIME IS RUNNING OUT TO CLAIM THIS FREE EBOOK 10:28:57

Learning Web Development with Bootstrap and AngularJS

Harness the power of both Bootstrap and AngularJS to build your own web app. Master the ideology of both frameworks and get hands-on as you build a contact manager throughout the course of the book. Discover filters, routing and views for your app. You will also learn how to connect your app to the server, customize Bootstrap and build a directive. Furthermore, you will look a potential stumbling blocks and will be given tips on how to avoid them. You will be building your dream web apps in no time.

NOT INTERESTED IN TODAY'S FREE EBOOK? THEN WHY NOT CHECK OUT ONE OF OUR PERMANENTLY FREE TITLES!

Free Big Data & Data Management eBooks

Hadoop Explained
Practical Data Analysis
What you need to know about R
MySQL Database
SQL Server Integration Services Using Visual Studio 2005
R Deep Learning Essentials

Free Machine Learning eBooks

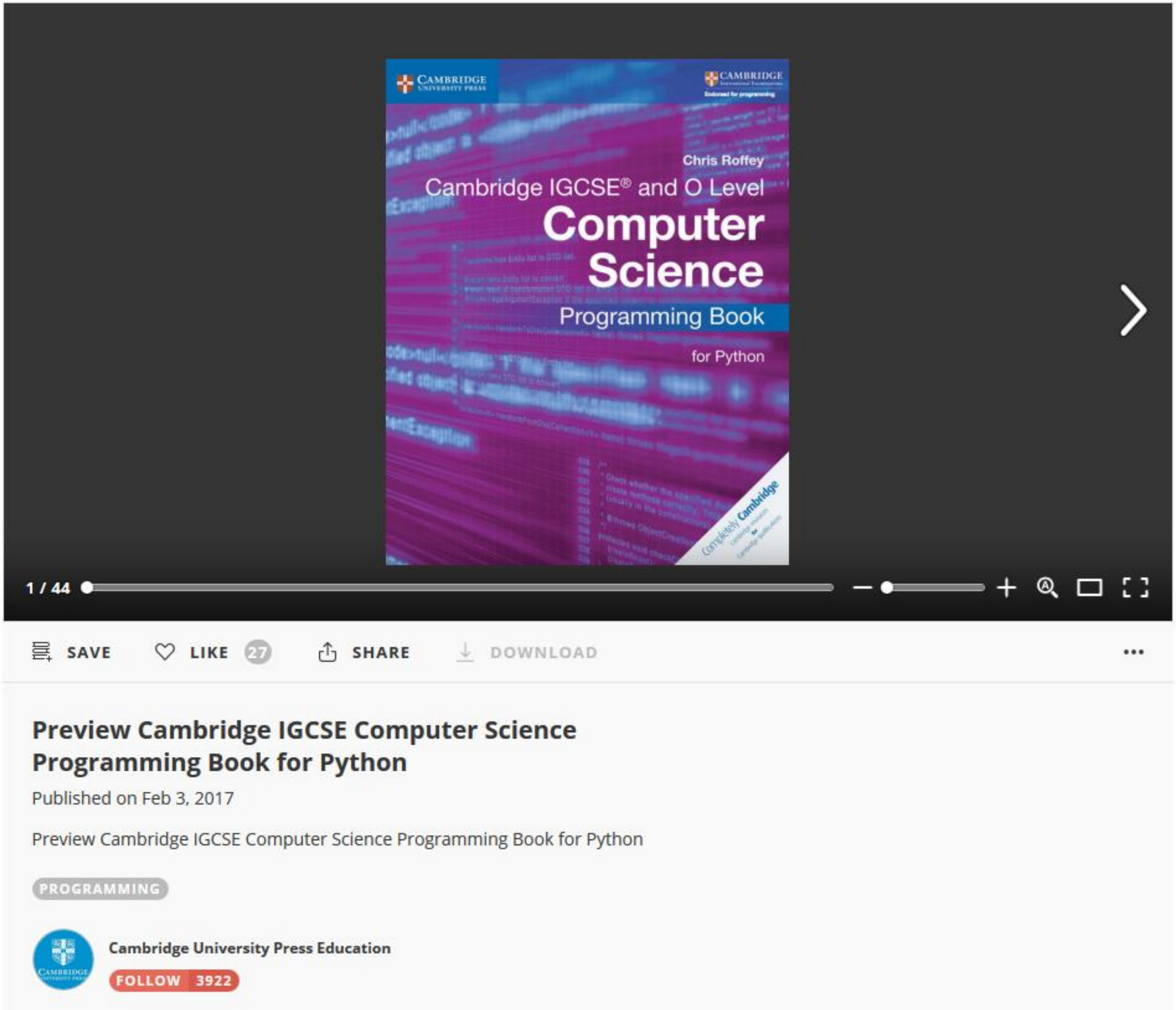
Building Machine Learning Systems with Python
What you Need to Know about Machine Learning
Machine Learning with R
Neural Network Programming with Java
Advanced Machine Learning with Python
Getting Started with TensorFlow

Free Python Programming eBooks

Learning Python
Mastering Python
What you need to know about Python
Advanced Machine Learning with Python
Learning OpenCV 3 Computer Vision with Python Second Edition
Raspberry Pi cookbook for Python programmers

Kingston University London

General Python Books



Preview Cambridge IGCSE Computer Science Programming Book for Python

Published on Feb 3, 2017

Preview Cambridge IGCSE Computer Science Programming Book for Python

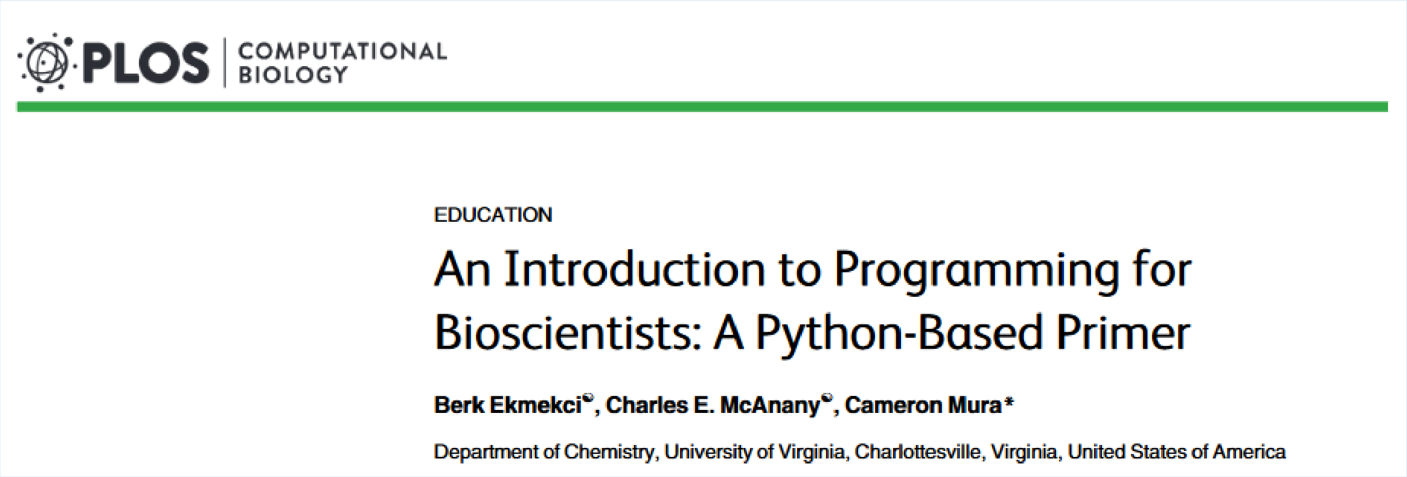
PROGRAMMING

Cambridge University Press Education

FOLLOW 3922

Kingston University London

Better Than a Book



PLOS | COMPUTATIONAL BIOLOGY

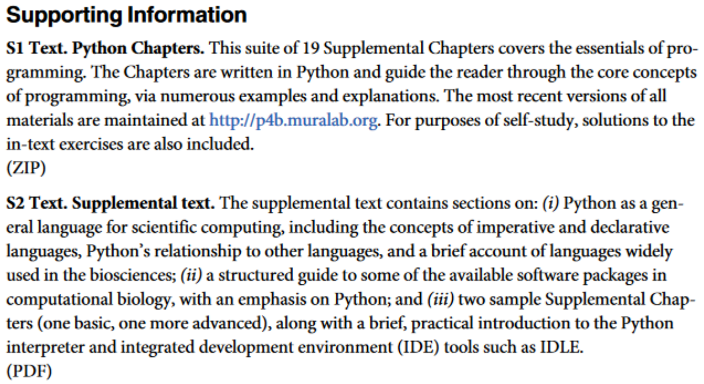
EDUCATION

An Introduction to Programming for Bioscientists: A Python-Based Primer

Berk Ekmekci[a], Charles E. McAnany[a], Cameron Mura*

Department of Chemistry, University of Virginia, Charlottesville, Virginia, United States of America

Supplementary Material



Supporting Information

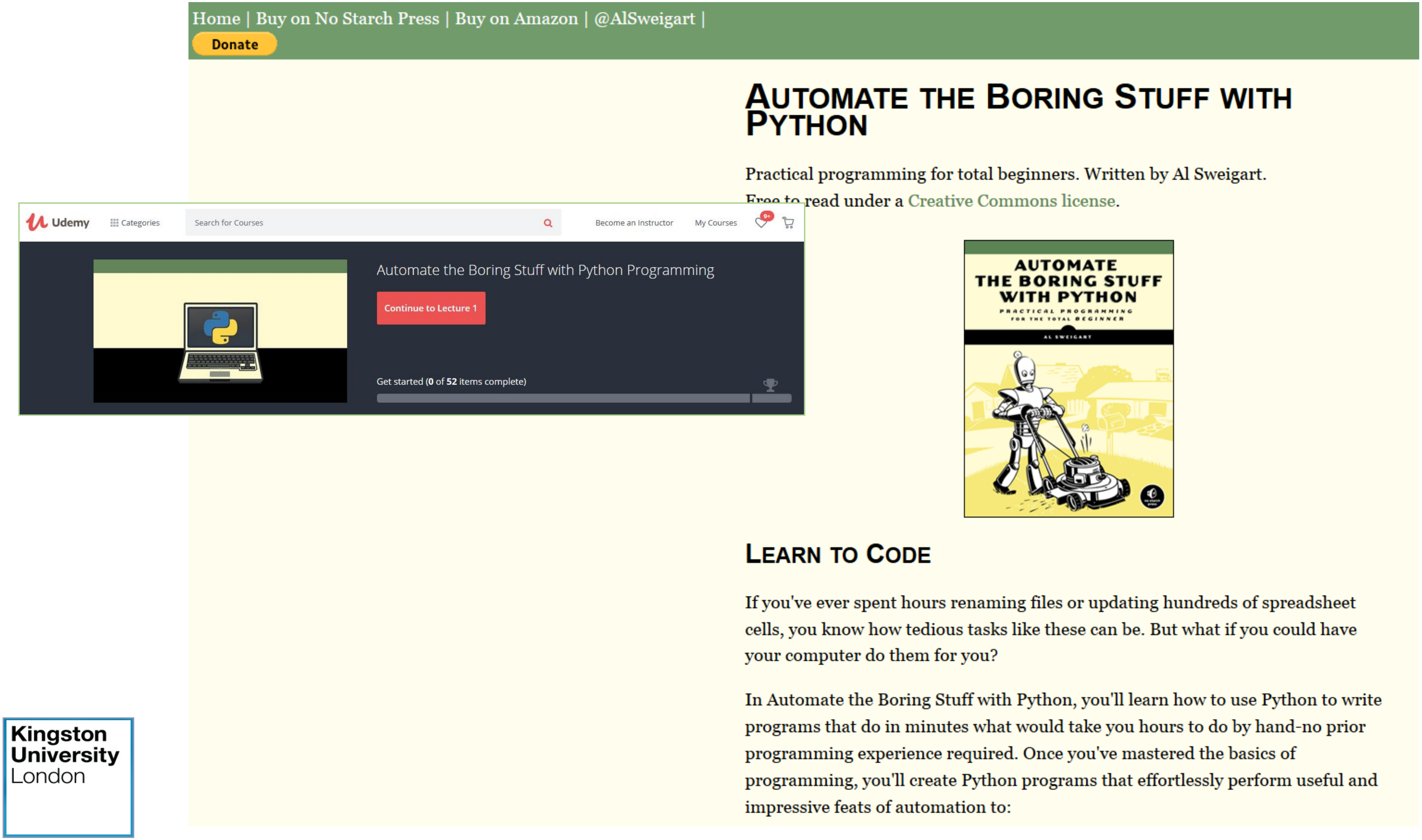
S1 Text. Python Chapters. This suite of 19 Supplemental Chapters covers the essentials of programming. The Chapters are written in Python and guide the reader through the core concepts of programming, via numerous examples and explanations. The most recent versions of all materials are maintained at http://p4b.muralab.org. For purposes of self-study, solutions to the in-text exercises are also included.
(ZIP)

S2 Text. Supplemental text. The supplemental text contains sections on: *(i)* Python as a general language for scientific computing, including the concepts of imperative and declarative languages, Python's relationship to other languages, and a brief account of languages widely used in the biosciences; *(ii)* a structured guide to some of the available software packages in computational biology, with an emphasis on Python; and *(iii)* two sample Supplemental Chapters (one basic, one more advanced), along with a brief, practical introduction to the Python interpreter and integrated development environment (IDE) tools such as IDLE.
(PDF)

Click above for the latest P4B materials (code chapters).

Kingston
University
London

Python: Online Learning


Home | Buy on No Starch Press | Buy on Amazon | @AlSweigart |
Donate
Udemy
Categories
Search for Courses
Become an Instructor
My Courses
9+
Continue to Lecture 1
AUTOMATE THE BORING STUFF WITH PYTHON
PRACTICAL PROGRAMMING FOR THE TOTAL BEGINNER
AL SWEIGART

Automate the Boring Stuff with Python Programming

Get started (**0** of **52** items complete)

Automate the Boring Stuff with Python

Practical programming for total beginners. Written by Al Sweigart. Free to read under a Creative Commons license.

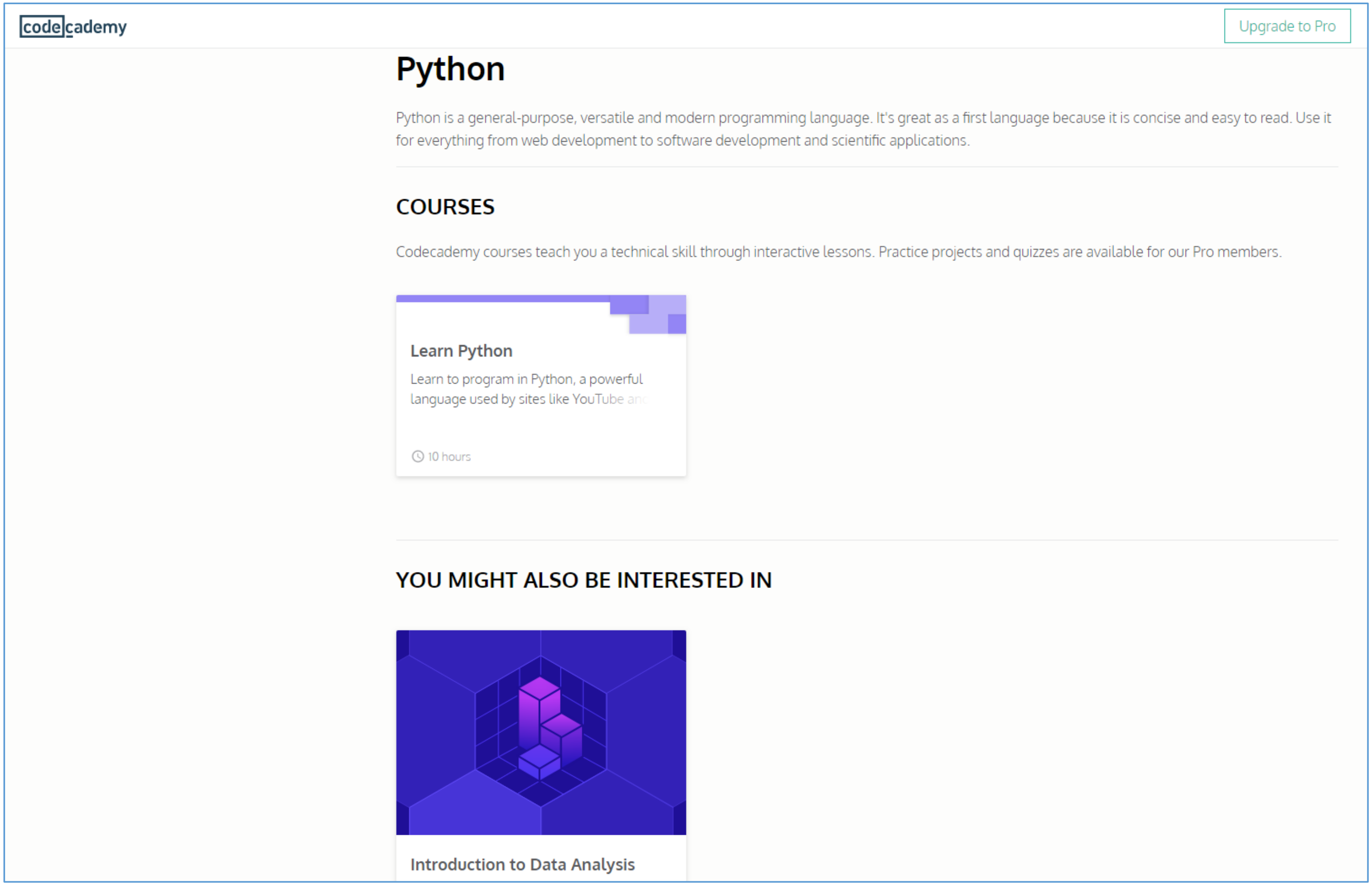
Learn to Code

If you've ever spent hours renaming files or updating hundreds of spreadsheet cells, you know how tedious tasks like these can be. But what if you could have your computer do them for you?

In Automate the Boring Stuff with Python, you'll learn how to use Python to write programs that do in minutes what would take you hours to do by hand-no prior programming experience required. Once you've mastered the basics of programming, you'll create Python programs that effortlessly perform useful and impressive feats of automation to:

Python: Online Learning

Code Academy: Python Courses (Free)



Python

Python is a general-purpose, versatile and modern programming language. It's great as a first language because it is concise and easy to read. Use it for everything from web development to software development and scientific applications.

COURSES

Codecademy courses teach you a technical skill through interactive lessons. Practice projects and quizzes are available for our Pro members.

Learn Python

Learn to program in Python, a powerful language used by sites like YouTube an

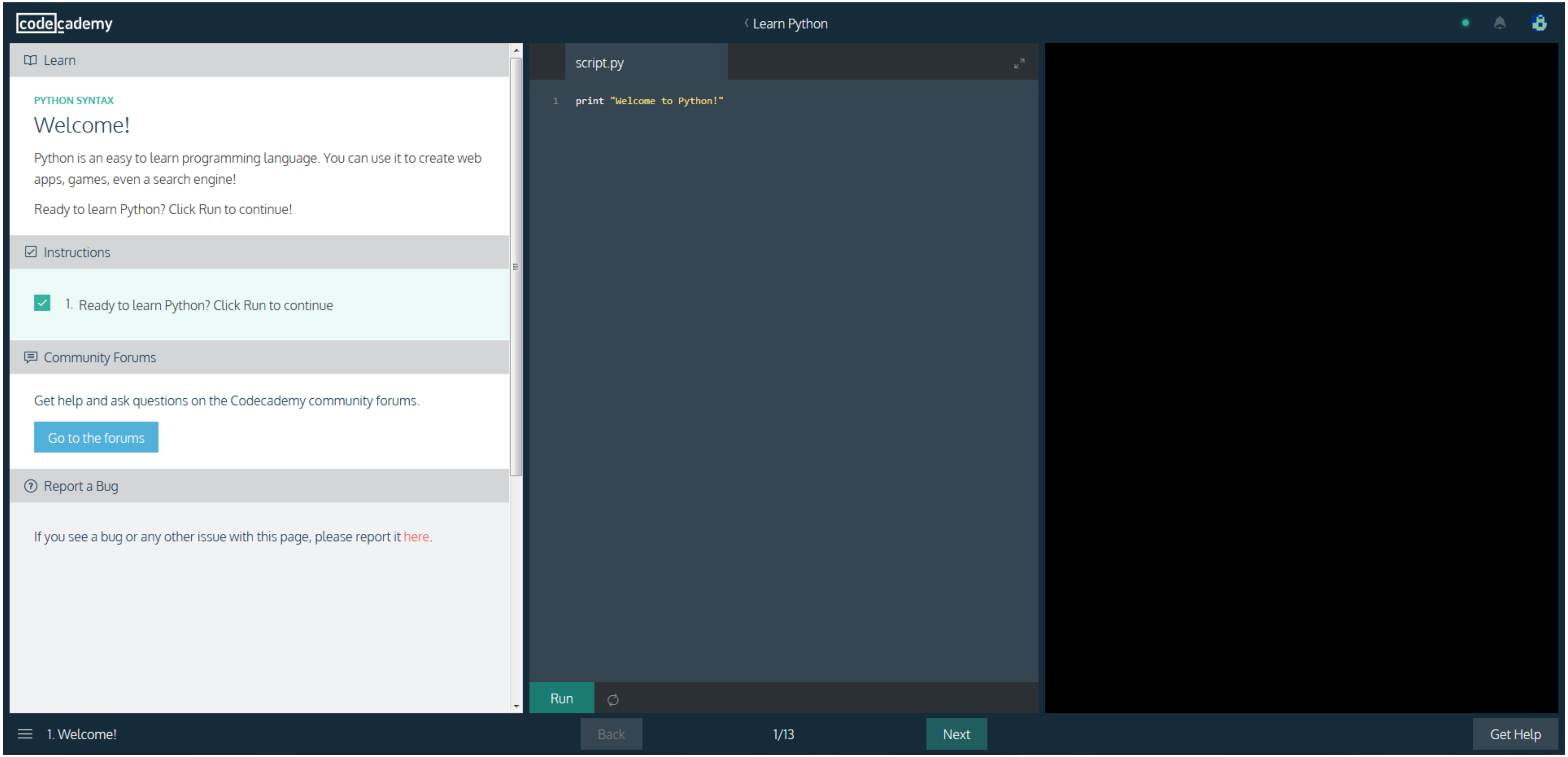
10 hours

YOU MIGHT ALSO BE INTERESTED IN

Introduction to Data Analysis

Python: Online Learning

Code Academy: Python Courses (Free)



PYTHON SYNTAX

Welcome!

Python is an easy to learn programming language. You can use it to create web apps, games, even a search engine!

Ready to learn Python? Click Run to continue!

☑ 1. Ready to learn Python? Click Run to continue

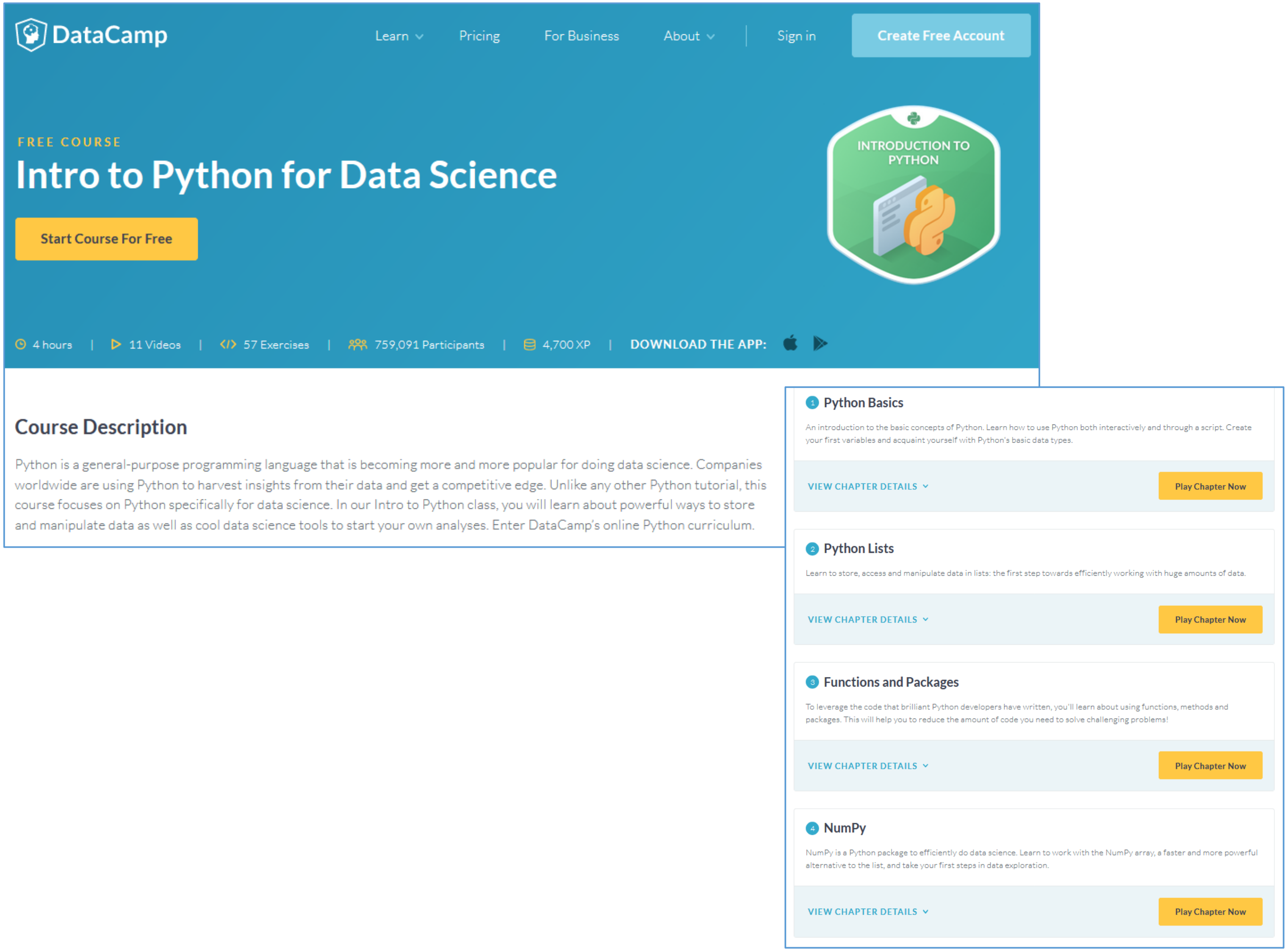
Get help and ask questions on the Codecademy community forums.

If you see a bug or any other issue with this page, please report it here.

```
print "Welcome to Python!"
```

Python: Online Learning

DataCamp: Intro to Python Course (Free)



FREE COURSE

Intro to Python for Data Science

Start Course For Free

4 hours | 11 Videos | 57 Exercises | 759,091 Participants | 4,700 XP | DOWNLOAD THE APP:

Course Description

Python is a general-purpose programming language that is becoming more and more popular for doing data science. Companies worldwide are using Python to harvest insights from their data and get a competitive edge. Unlike any other Python tutorial, this course focuses on Python specifically for data science. In our Intro to Python class, you will learn about powerful ways to store and manipulate data as well as cool data science tools to start your own analyses. Enter DataCamp's online Python curriculum.

1 Python Basics

An introduction to the basic concepts of Python. Learn how to use Python both interactively and through a script. Create your first variables and acquaint yourself with Python's basic data types.

VIEW CHAPTER DETAILS ⌄ Play Chapter Now

2 Python Lists

Learn to store, access and manipulate data in lists: the first step towards efficiently working with huge amounts of data.

VIEW CHAPTER DETAILS ⌄ Play Chapter Now

3 Functions and Packages

To leverage the code that brilliant Python developers have written, you'll learn about using functions, methods and packages. This will help you to reduce the amount of code you need to solve challenging problems!

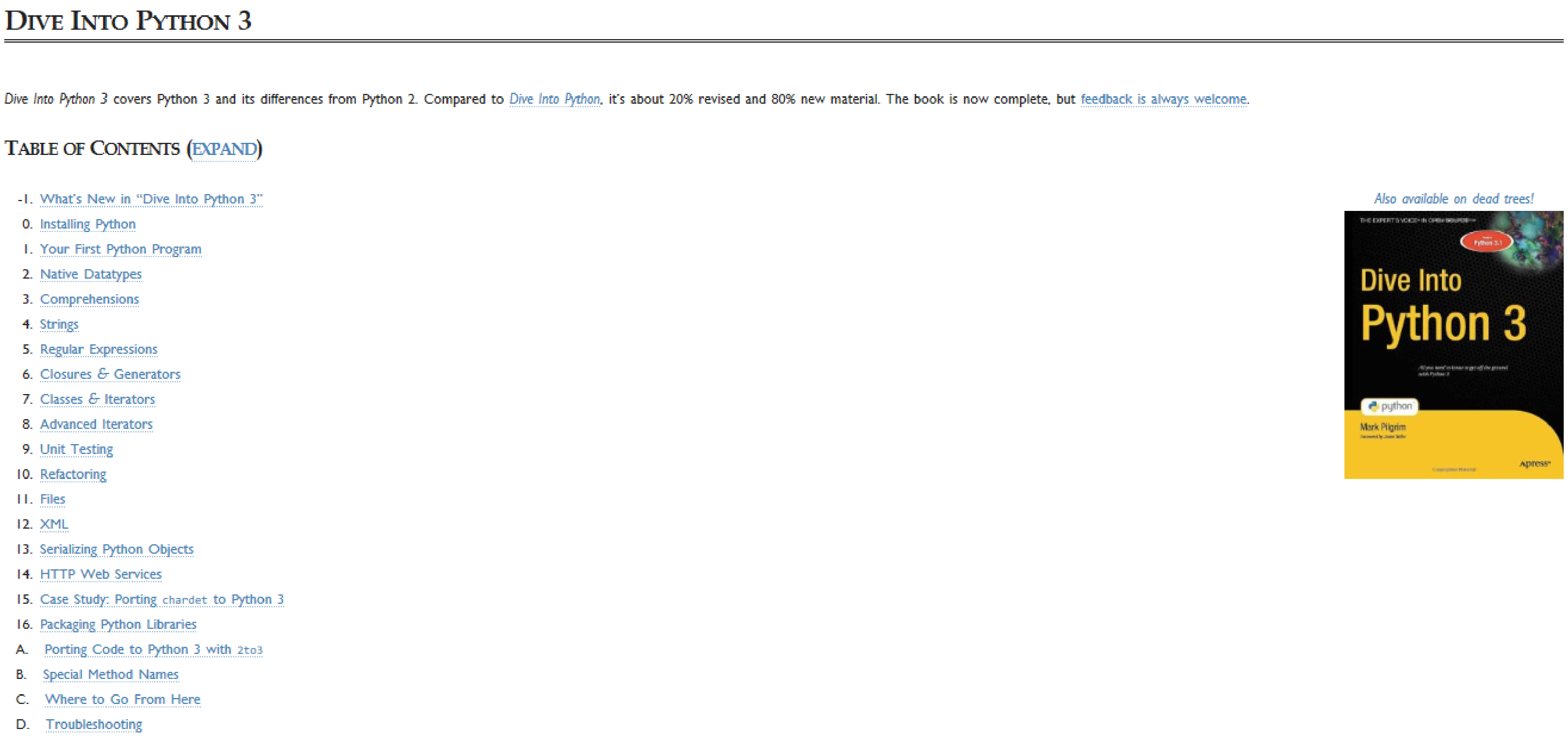
VIEW CHAPTER DETAILS ⌄ Play Chapter Now

4 NumPy

NumPy is a Python package to efficiently do data science. Learn to work with the NumPy array, a faster and more powerful alternative to the list, and take your first steps in data exploration.

VIEW CHAPTER DETAILS ⌄ Play Chapter Now

Python: Online Learning



Dive Into Python 3

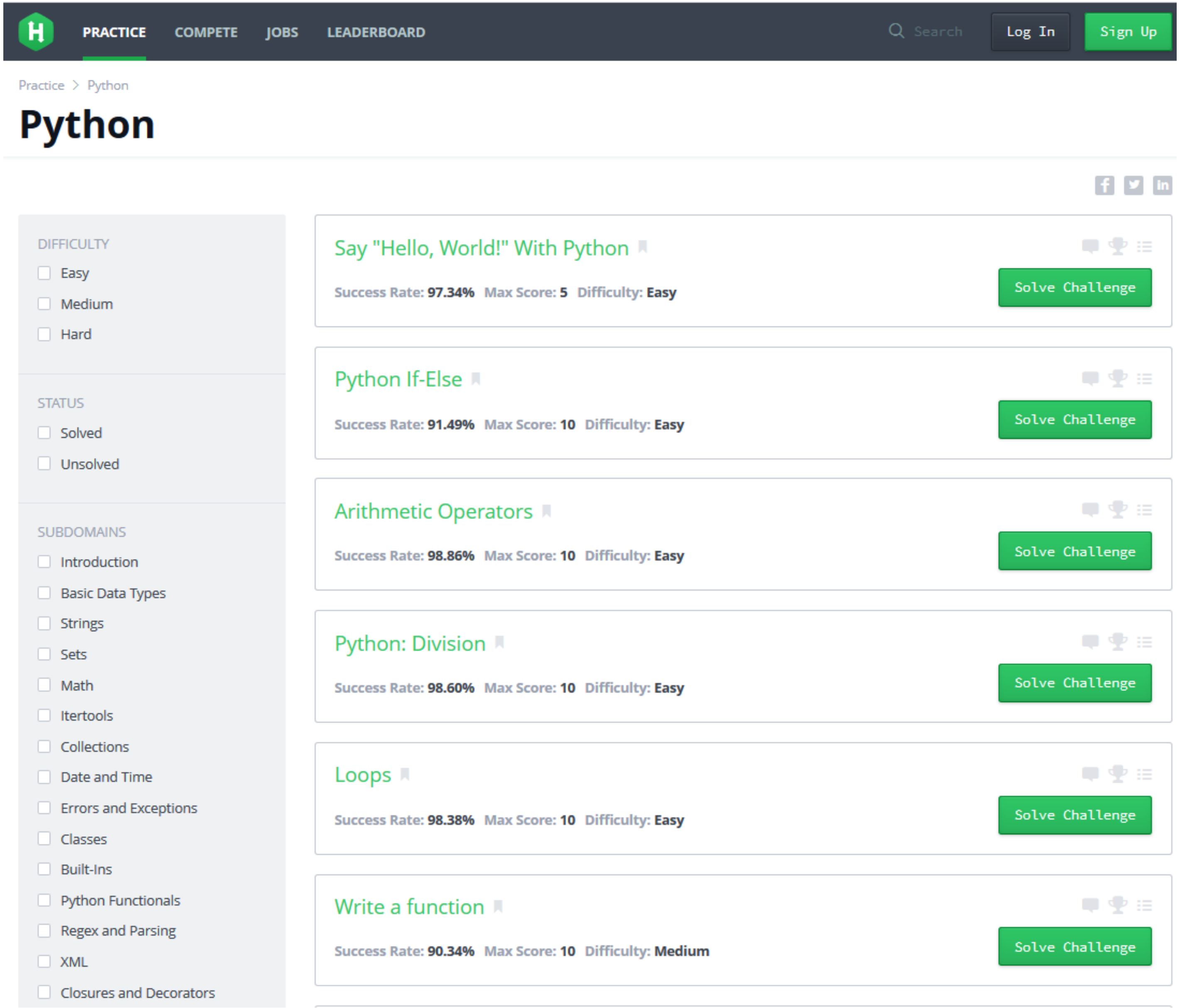
Dive Into Python 3 covers Python 3 and its differences from Python 2. Compared to *Dive Into Python*, it's about 20% revised and 80% new material. The book is now complete, but feedback is always welcome.

Table of Contents (expand)

-1. What's New in "Dive Into Python 3"
0. Installing Python
1. Your First Python Program
2. Native Datatypes
3. Comprehensions
4. Strings
5. Regular Expressions
6. Closures & Generators
7. Classes & Iterators
8. Advanced Iterators
9. Unit Testing
10. Refactoring
11. Files
12. XML
13. Serializing Python Objects
14. HTTP Web Services
15. Case Study: Porting `chardet` to Python 3
16. Packaging Python Libraries
A. Porting Code to Python 3 with `2to3`
B. Special Method Names
C. Where to Go From Here
D. Troubleshooting

Also available on dead trees!

Python: Online Learning



Practice > Python

Python

DIFFICULTY

□ Easy
□ Medium
□ Hard

STATUS

□ Solved
□ Unsolved

SUBDOMAINS

□ Introduction
□ Basic Data Types
□ Strings
□ Sets
□ Math
□ Itertools
□ Collections
□ Date and Time
□ Errors and Exceptions
□ Classes
□ Built-Ins
□ Python Functionals
□ Regex and Parsing
□ XML
□ Closures and Decorators

Say "Hello, World!" With Python

Success Rate: 97.34% Max Score: 5 Difficulty: Easy

Solve Challenge

Python If-Else

Success Rate: 91.49% Max Score: 10 Difficulty: Easy

Solve Challenge

Arithmetic Operators

Success Rate: 98.86% Max Score: 10 Difficulty: Easy

Solve Challenge

Python: Division

Success Rate: 98.60% Max Score: 10 Difficulty: Easy

Solve Challenge

Loops

Success Rate: 98.38% Max Score: 10 Difficulty: Easy

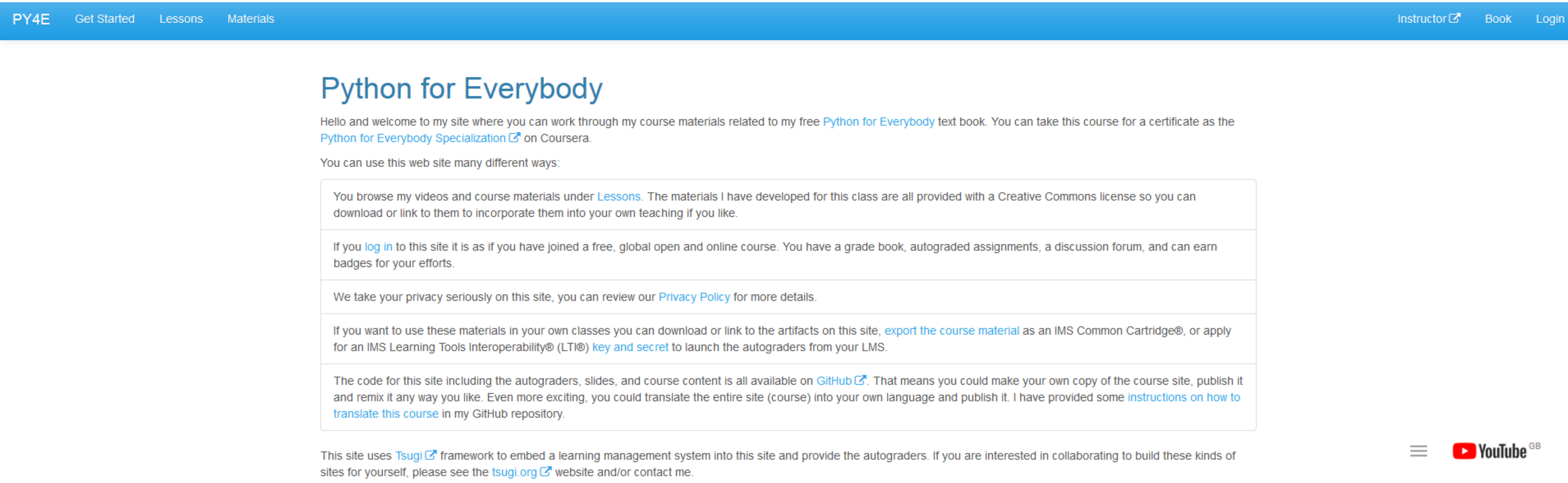
Solve Challenge

Write a function

Success Rate: 90.34% Max Score: 10 Difficulty: Medium

Solve Challenge

Python: Online Learning



Python for Everybody

Hello and welcome to my site where you can work through my course materials related to my free Python for Everybody text book. You can take this course for a certificate as the Python for Everybody Specialization on Coursera.

You can use this web site many different ways:

You browse my videos and course materials under Lessons. The materials I have developed for this class are all provided with a Creative Commons license so you can download or link to them to incorporate them into your own teaching if you like.

If you log in to this site it is as if you have joined a free, global open and online course. You have a grade book, autograded assignments, a discussion forum, and can earn badges for your efforts.

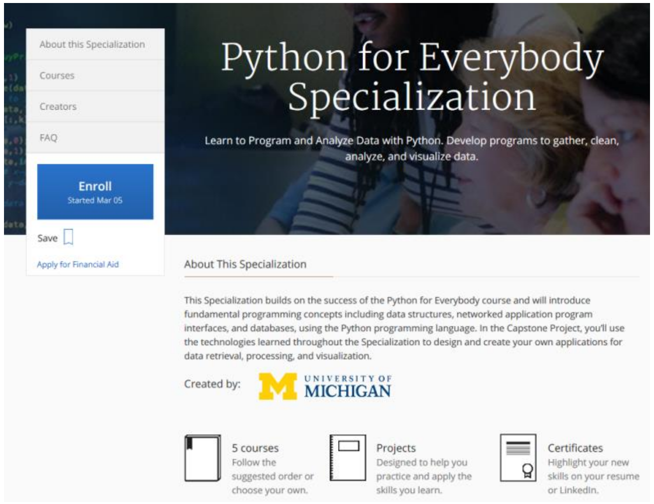
We take your privacy seriously on this site, you can review our Privacy Policy for more details.

If you want to use these materials in your own classes you can download or link to the artifacts on this site, export the course material as an IMS Common Cartridge®, or apply for an IMS Learning Tools Interoperability® (LTI®) key and secret to launch the autograders from your LMS.

The code for this site including the autograders, slides, and course content is all available on GitHub. That means you could make your own copy of the course site, publish it and remix it any way you like. Even more exciting, you could translate the entire site (course) into your own language and publish it. I have provided some instructions on how to translate this course in my GitHub repository.

This site uses Tsugi framework to embed a learning management system into this site and provide the autograders. If you are interested in collaborating to build these kinds of sites for yourself, please see the tsugi.org website and/or contact me.

coursera



Python for Everybody Specialization

Learn to Program and Analyze Data with Python. Develop programs to gather, clean, analyze, and visualize data.

About This Specialization

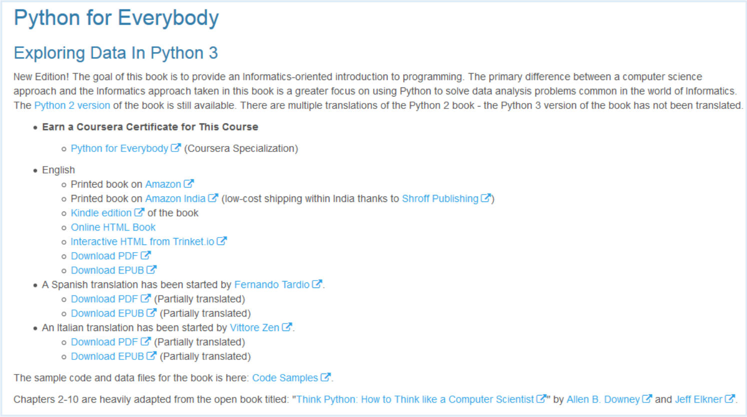
This Specialization builds on the success of the Python for Everybody course and will introduce fundamental programming concepts including data structures, networked application program interfaces, and databases, using the Python programming language. In the Capstone Project, you'll use the technologies learned throughout the Specialization to design and create your own applications for data retrieval, processing, and visualization.

Created by: UNIVERSITY OF MICHIGAN

5 courses
Follow the suggested order or choose your own.

Projects
Designed to help you practice and apply the skills you learn.

Certificates
Highlight your new skills on your resume or LinkedIn.



Python for Everybody

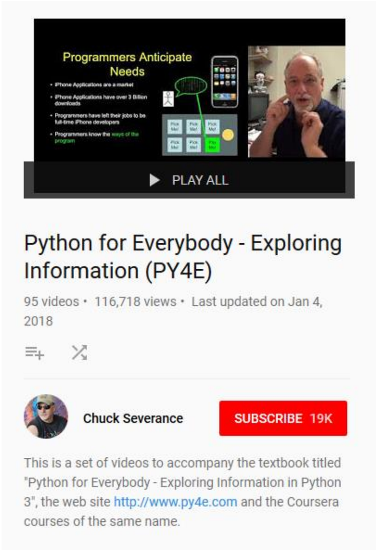
Exploring Data In Python 3

New Edition! The goal of this book is to provide an Informatics-oriented introduction to programming. The primary difference between a computer science approach and the Informatics approach taken in this book is a greater focus on using Python to solve data analysis problems common in the world of Informatics. The Python 2 version of the book is still available. There are multiple translations of the Python 2 book - the Python 3 version of the book has not been translated.

- **Earn a Coursera Certificate for This Course**
    - Python for Everybody (Coursera Specialization)
- English
    - Printed book on Amazon
    - Printed book on Amazon India (low-cost shipping within India thanks to Shroff Publishing)
    - Kindle edition of the book
    - Online HTML Book
    - Interactive HTML from Trinket.io
    - Download PDF
    - Download EPUB
- A Spanish translation has been started by Fernando Tardio.
    - Download PDF (Partially translated)
    - Download EPUB (Partially translated)
- An Italian translation has been started by Vittore Zen.
    - Download PDF (Partially translated)
    - Download EPUB (Partially translated)

The sample code and data files for the book is here: Code Samples.

Chapters 2-10 are heavily adapted from the open book titled: "Think Python: How to Think like a Computer Scientist" by Allen B. Downey and Jeff Elkner.



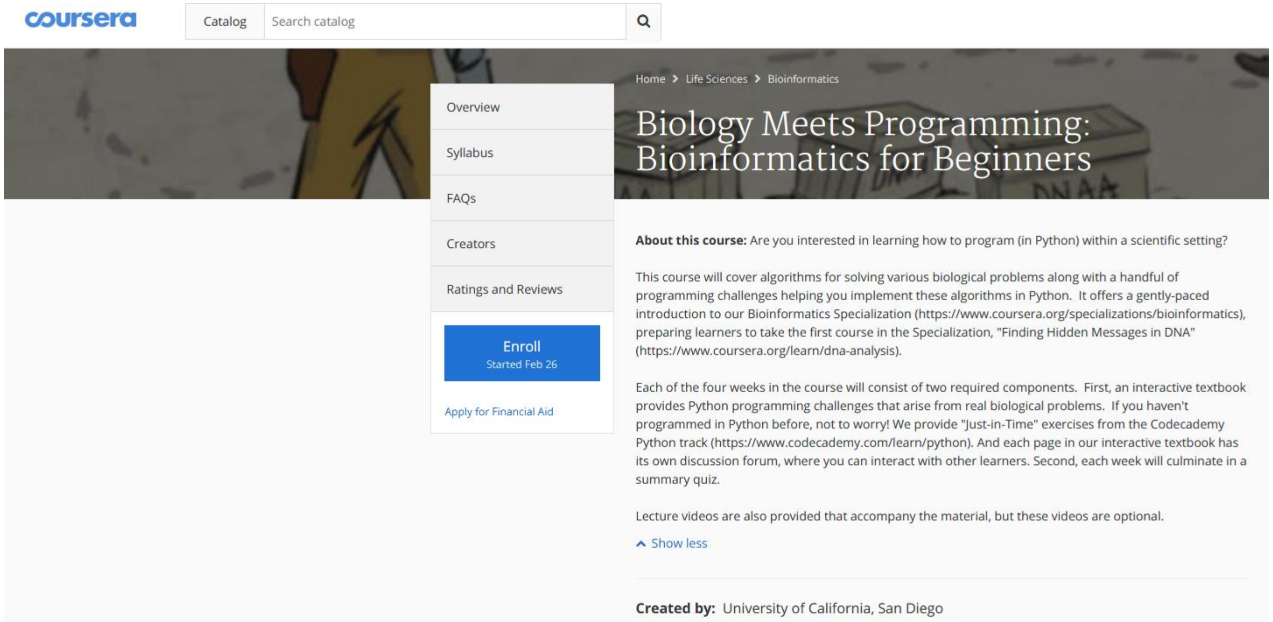
Python for Everybody - Exploring Information (PY4E)

95 videos • 116,718 views • Last updated on Jan 4, 2018

This is a set of videos to accompany the textbook titled "Python for Everybody - Exploring Information in Python 3", the web site http://www.py4e.com and the Coursera courses of the same name.

Kingston University London

Python: Online Learning



Biology Meets Programming: Bioinformatics for Beginners

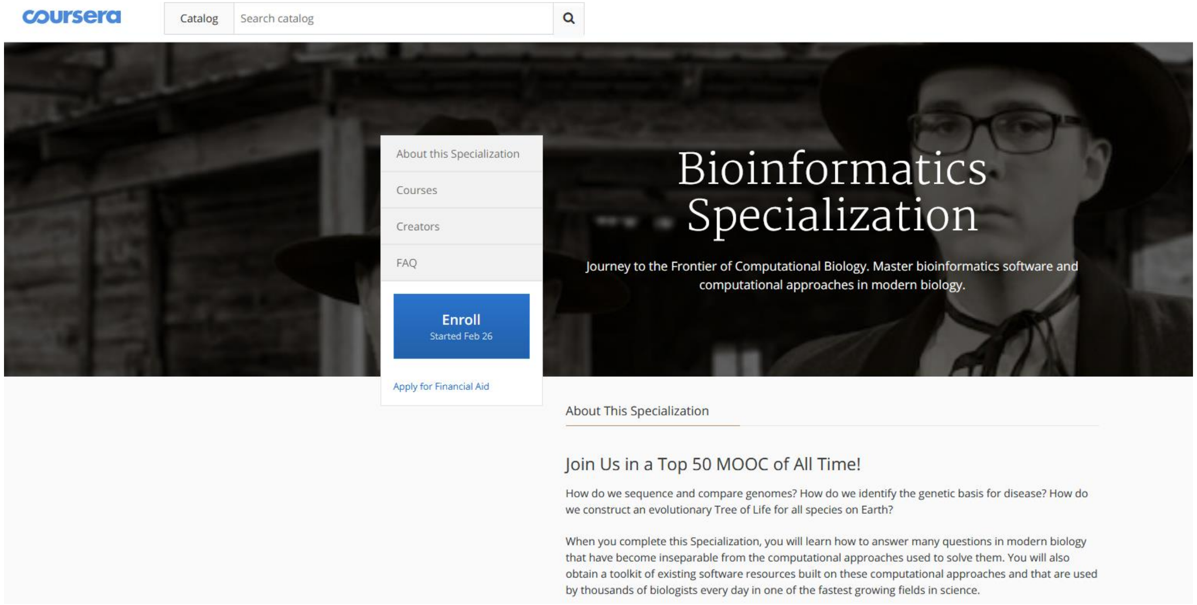
About this course: Are you interested in learning how to program (in Python) within a scientific setting?

This course will cover algorithms for solving various biological problems along with a handful of programming challenges helping you implement these algorithms in Python. It offers a gently-paced introduction to our Bioinformatics Specialization (https://www.coursera.org/specializations/bioinformatics), preparing learners to take the first course in the Specialization, "Finding Hidden Messages in DNA" (https://www.coursera.org/learn/dna-analysis).

Each of the four weeks in the course will consist of two required components. First, an interactive textbook provides Python programming challenges that arise from real biological problems. If you haven't programmed in Python before, not to worry! We provide "Just-in-Time" exercises from the Codecademy Python track (https://www.codecademy.com/learn/python). And each page in our interactive textbook has its own discussion forum, where you can interact with other learners. Second, each week will culminate in a summary quiz.

Lecture videos are also provided that accompany the material, but these videos are optional.

Created by: University of California, San Diego



Bioinformatics Specialization

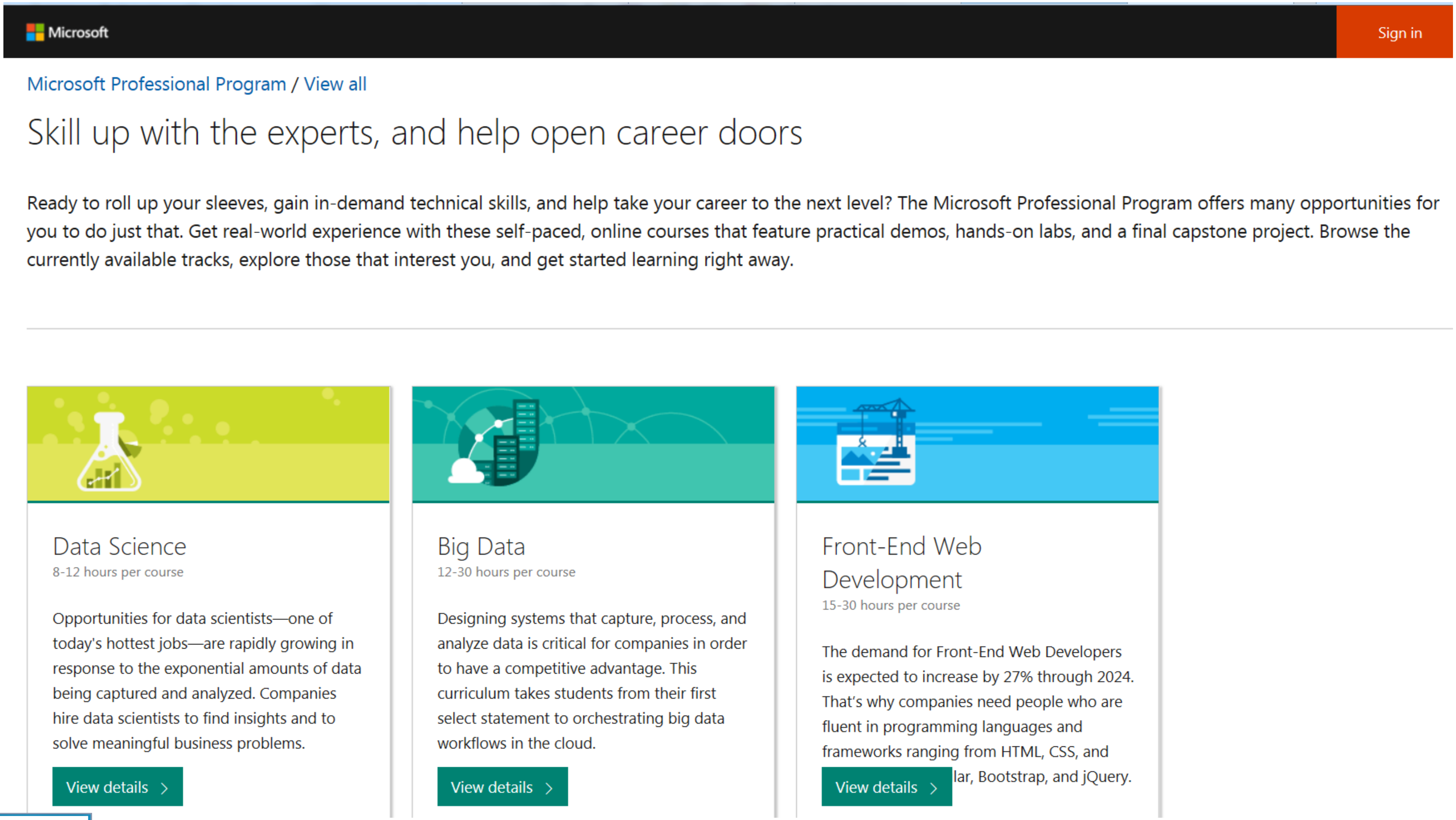
Journey to the Frontier of Computational Biology. Master bioinformatics software and computational approaches in modern biology.

Join Us in a Top 50 MOOC of All Time!

How do we sequence and compare genomes? How do we identify the genetic basis for disease? How do we construct an evolutionary Tree of Life for all species on Earth?

When you complete this Specialization, you will learn how to answer many questions in modern biology that have become inseparable from the computational approaches used to solve them. You will also obtain a toolkit of existing software resources built on these computational approaches and that are used by thousands of biologists every day in one of the fastest growing fields in science.

Python: Online Learning



Microsoft Professional Program / View all

Skill up with the experts, and help open career doors

Ready to roll up your sleeves, gain in-demand technical skills, and help take your career to the next level? The Microsoft Professional Program offers many opportunities for you to do just that. Get real-world experience with these self-paced, online courses that feature practical demos, hands-on labs, and a final capstone project. Browse the currently available tracks, explore those that interest you, and get started learning right away.

Data Science

8-12 hours per course

Opportunities for data scientists—one of today's hottest jobs—are rapidly growing in response to the exponential amounts of data being captured and analyzed. Companies hire data scientists to find insights and to solve meaningful business problems.

Big Data

12-30 hours per course

Designing systems that capture, process, and analyze data is critical for companies in order to have a competitive advantage. This curriculum takes students from their first select statement to orchestrating big data workflows in the cloud.

Front-End Web Development

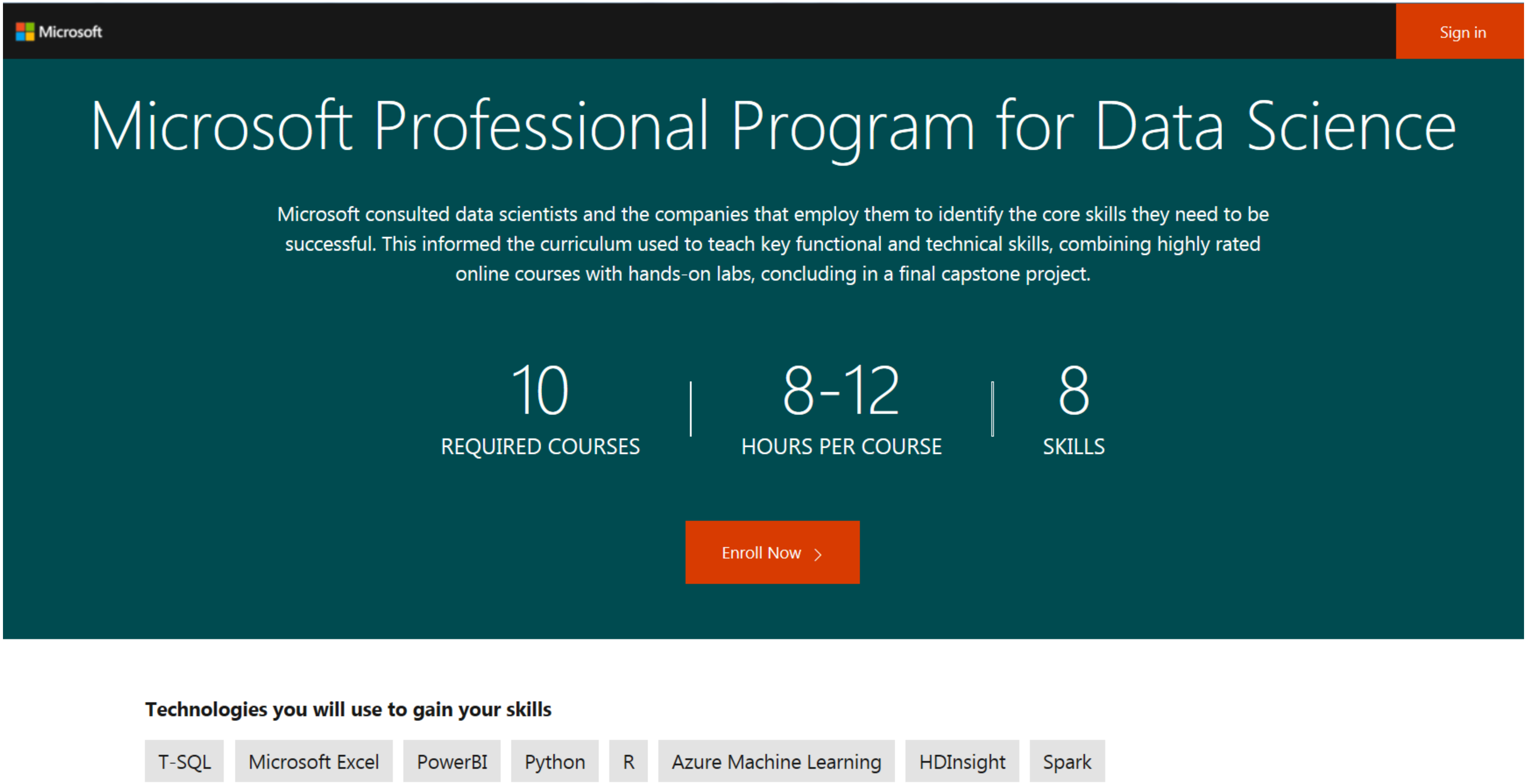
15-30 hours per course

The demand for Front-End Web Developers is expected to increase by 27% through 2024. That's why companies need people who are fluent in programming languages and frameworks ranging from HTML, CSS, and

lar, Bootstrap, and jQuery.

Kingston University London

Python: Online Learning



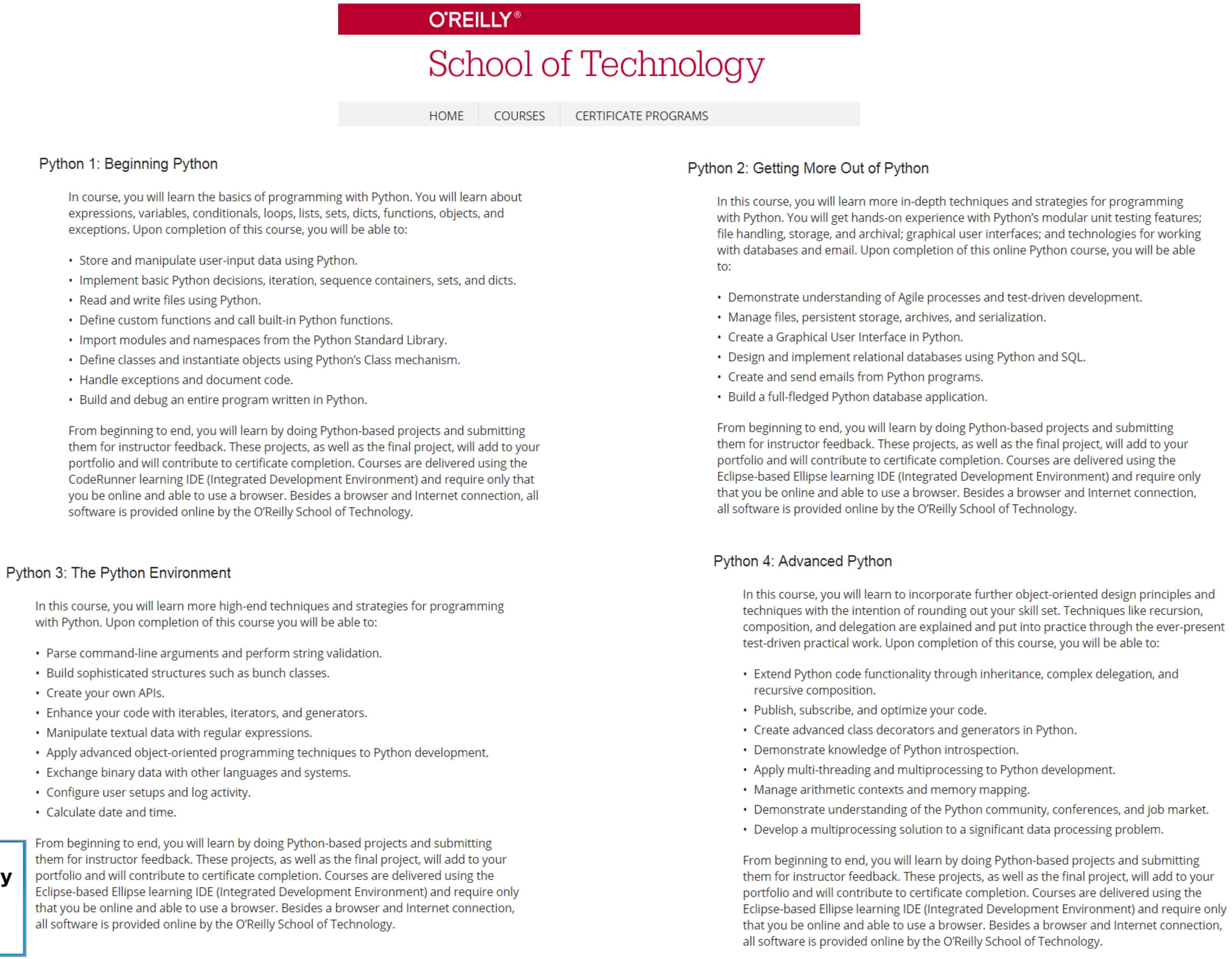
Microsoft Professional Program for Data Science

Microsoft consulted data scientists and the companies that employ them to identify the core skills they need to be successful. This informed the curriculum used to teach key functional and technical skills, combining highly rated online courses with hands-on labs, concluding in a final capstone project.

10 REQUIRED COURSES | 8-12 HOURS PER COURSE | 8 SKILLS

Technologies you will use to gain your skills

T-SQL | Microsoft Excel | PowerBI | Python | R | Azure Machine Learning | HDInsight | Spark



O'REILLY®

School of Technology

Python 1: Beginning Python

In course, you will learn the basics of programming with Python. You will learn about expressions, variables, conditionals, loops, lists, sets, dicts, functions, objects, and exceptions. Upon completion of this course, you will be able to:

- Store and manipulate user-input data using Python.
- Implement basic Python decisions, iteration, sequence containers, sets, and dicts.
- Read and write files using Python.
- Define custom functions and call built-in Python functions.
- Import modules and namespaces from the Python Standard Library.
- Define classes and instantiate objects using Python's Class mechanism.
- Handle exceptions and document code.
- Build and debug an entire program written in Python.

From beginning to end, you will learn by doing Python-based projects and submitting them for instructor feedback. These projects, as well as the final project, will add to your portfolio and will contribute to certificate completion. Courses are delivered using the CodeRunner learning IDE (Integrated Development Environment) and require only that you be online and able to use a browser. Besides a browser and Internet connection, all software is provided online by the O'Reilly School of Technology.

Python 2: Getting More Out of Python

In this course, you will learn more in-depth techniques and strategies for programming with Python. You will get hands-on experience with Python's modular unit testing features; file handling, storage, and archival; graphical user interfaces; and technologies for working with databases and email. Upon completion of this online Python course, you will be able to:

- Demonstrate understanding of Agile processes and test-driven development.
- Manage files, persistent storage, archives, and serialization.
- Create a Graphical User Interface in Python.
- Design and implement relational databases using Python and SQL.
- Create and send emails from Python programs.
- Build a full-fledged Python database application.

From beginning to end, you will learn by doing Python-based projects and submitting them for instructor feedback. These projects, as well as the final project, will add to your portfolio and will contribute to certificate completion. Courses are delivered using the Eclipse-based Ellipse learning IDE (Integrated Development Environment) and require only that you be online and able to use a browser. Besides a browser and Internet connection, all software is provided online by the O'Reilly School of Technology.

Python 3: The Python Environment

In this course, you will learn more high-end techniques and strategies for programming with Python. Upon completion of this course you will be able to:

- Parse command-line arguments and perform string validation.
- Build sophisticated structures such as bunch classes.
- Create your own APIs.
- Enhance your code with iterables, iterators, and generators.
- Manipulate textual data with regular expressions.
- Apply advanced object-oriented programming techniques to Python development.
- Exchange binary data with other languages and systems.
- Configure user setups and log activity.
- Calculate date and time.

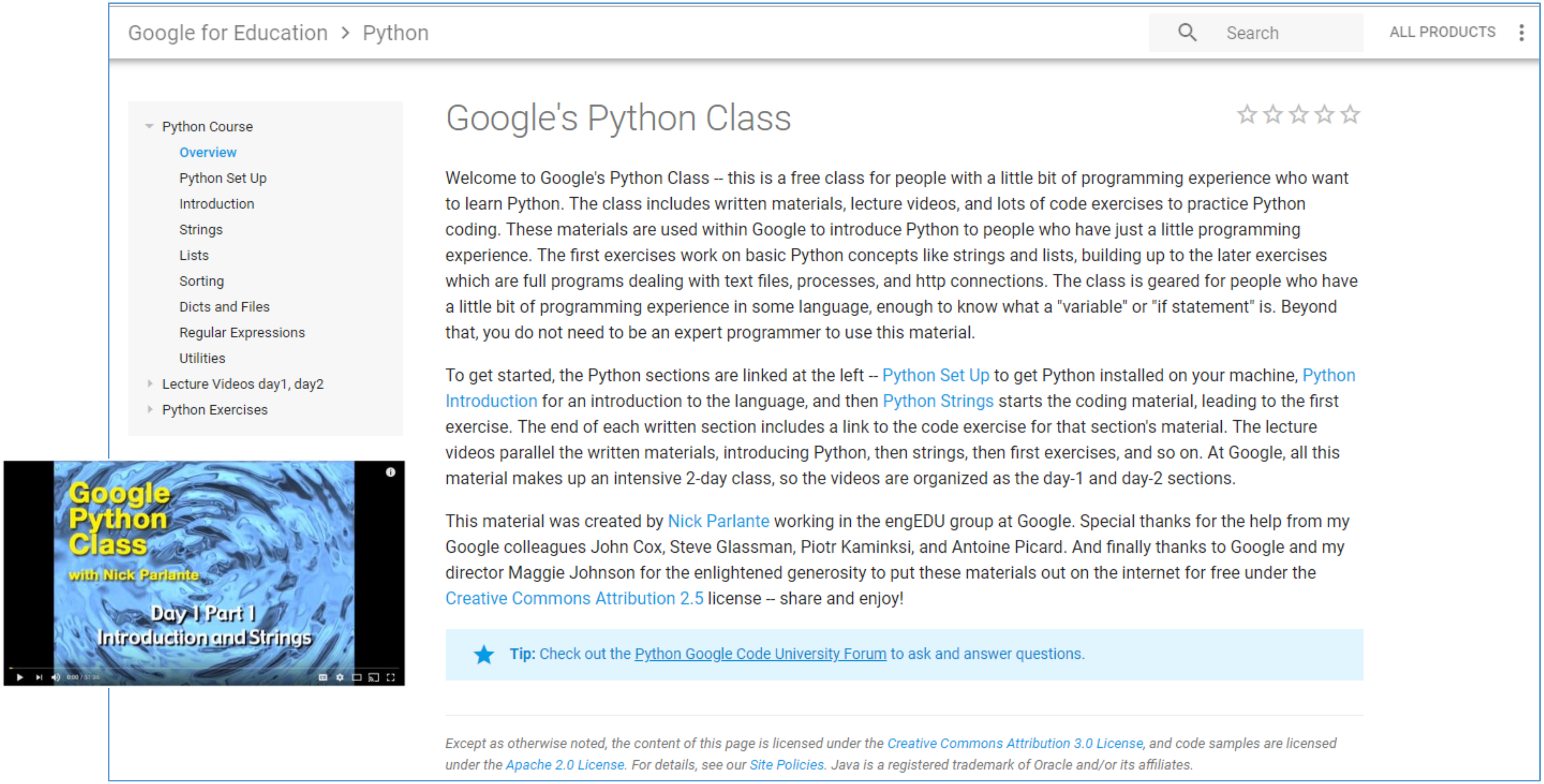
From beginning to end, you will learn by doing Python-based projects and submitting them for instructor feedback. These projects, as well as the final project, will add to your portfolio and will contribute to certificate completion. Courses are delivered using the Eclipse-based Ellipse learning IDE (Integrated Development Environment) and require only that you be online and able to use a browser. Besides a browser and Internet connection, all software is provided online by the O'Reilly School of Technology.

Python 4: Advanced Python

In this course, you will learn to incorporate further object-oriented design principles and techniques with the intention of rounding out your skill set. Techniques like recursion, composition, and delegation are explained and put into practice through the ever-present test-driven practical work. Upon completion of this course, you will be able to:

- Extend Python code functionality through inheritance, complex delegation, and recursive composition.
- Publish, subscribe, and optimize your code.
- Create advanced class decorators and generators in Python.
- Demonstrate knowledge of Python introspection.
- Apply multi-threading and multiprocessing to Python development.
- Manage arithmetic contexts and memory mapping.
- Demonstrate understanding of the Python community, conferences, and job market.
- Develop a multiprocessing solution to a significant data processing problem.

From beginning to end, you will learn by doing Python-based projects and submitting them for instructor feedback. These projects, as well as the final project, will add to your portfolio and will contribute to certificate completion. Courses are delivered using the Eclipse-based Ellipse learning IDE (Integrated Development Environment) and require only that you be online and able to use a browser. Besides a browser and Internet connection, all software is provided online by the O'Reilly School of Technology.



Google's Python Class

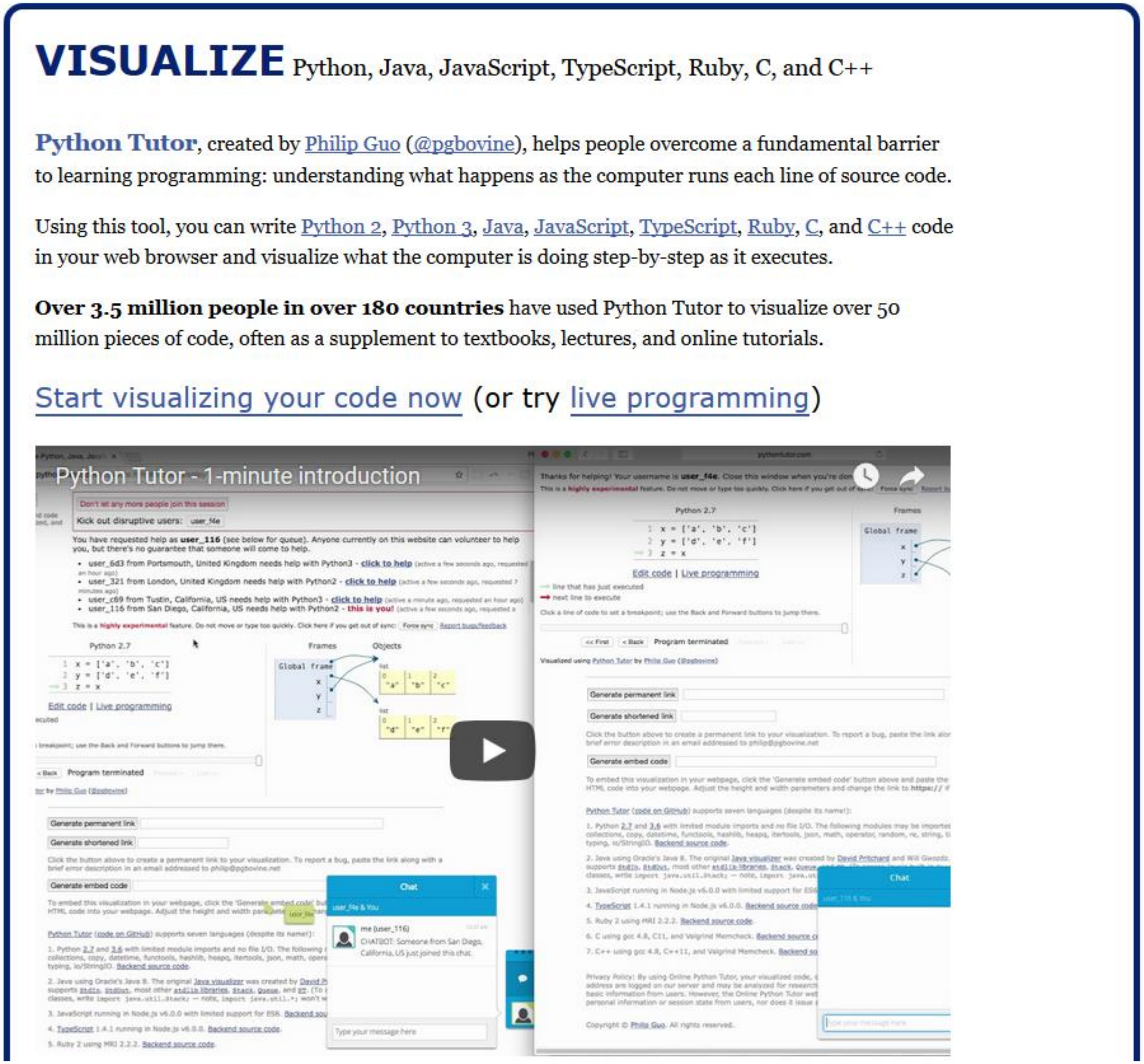
Welcome to Google's Python Class -- this is a free class for people with a little bit of programming experience who want to learn Python. The class includes written materials, lecture videos, and lots of code exercises to practice Python coding. These materials are used within Google to introduce Python to people who have just a little programming experience. The first exercises work on basic Python concepts like strings and lists, building up to the later exercises which are full programs dealing with text files, processes, and http connections. The class is geared for people who have a little bit of programming experience in some language, enough to know what a "variable" or "if statement" is. Beyond that, you do not need to be an expert programmer to use this material.

To get started, the Python sections are linked at the left -- Python Set Up to get Python installed on your machine, Python Introduction for an introduction to the language, and then Python Strings starts the coding material, leading to the first exercise. The end of each written section includes a link to the code exercise for that section's material. The lecture videos parallel the written materials, introducing Python, then strings, then first exercises, and so on. At Google, all this material makes up an intensive 2-day class, so the videos are organized as the day-1 and day-2 sections.

This material was created by Nick Parlante working in the engEDU group at Google. Special thanks for the help from my Google colleagues John Cox, Steve Glassman, Piotr Kaminksi, and Antoine Picard. And finally thanks to Google and my director Maggie Johnson for the enlightened generosity to put these materials out on the internet for free under the Creative Commons Attribution 2.5 license -- share and enjoy!

★ **Tip:** Check out the Python Google Code University Forum to ask and answer questions.

Online Help: Python Tutor



VISUALIZE Python, Java, JavaScript, TypeScript, Ruby, C, and C++

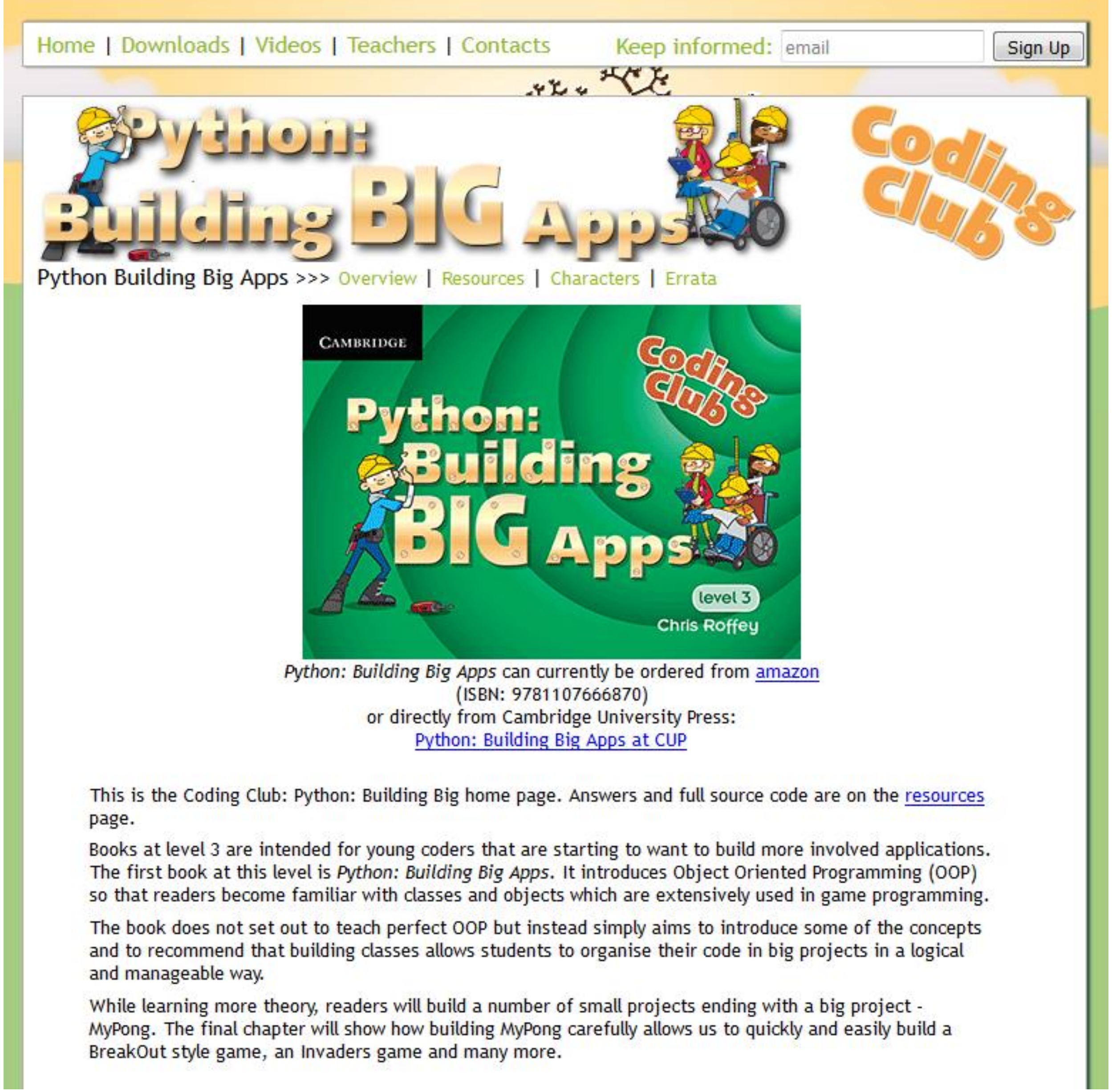
Python Tutor, created by Philip Guo (@pgbovine), helps people overcome a fundamental barrier to learning programming: understanding what happens as the computer runs each line of source code.

Using this tool, you can write Python 2, Python 3, Java, JavaScript, TypeScript, Ruby, C, and C++ code in your web browser and visualize what the computer is doing step-by-step as it executes.

Over 3.5 million people in over 180 countries have used Python Tutor to visualize over 50 million pieces of code, often as a supplement to textbooks, lectures, and online tutorials.

Start visualizing your code now (or try live programming)

Python for Fun


Home | Downloads | Videos | Teachers | Contacts
Keep informed: email Sign Up
Python: Building BIG Apps
Coding Club
Python Building Big Apps >>> Overview | Resources | Characters | Errata
CAMBRIDGE
Python: Building BIG Apps
Coding Club
level 3
Chris Roffey

Python: Building Big Apps can currently be ordered from amazon
(ISBN: 9781107666870)
or directly from Cambridge University Press:
Python: Building Big Apps at CUP

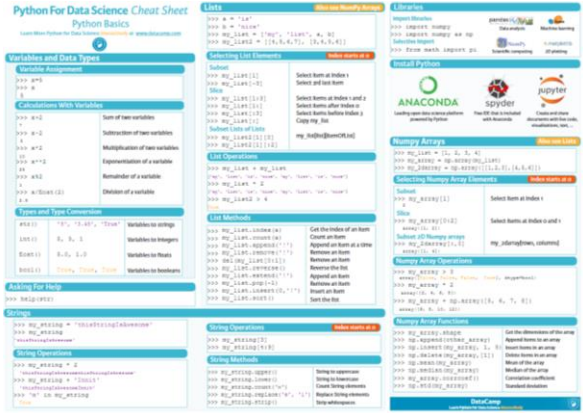
This is the Coding Club: Python: Building Big home page. Answers and full source code are on the resources page.

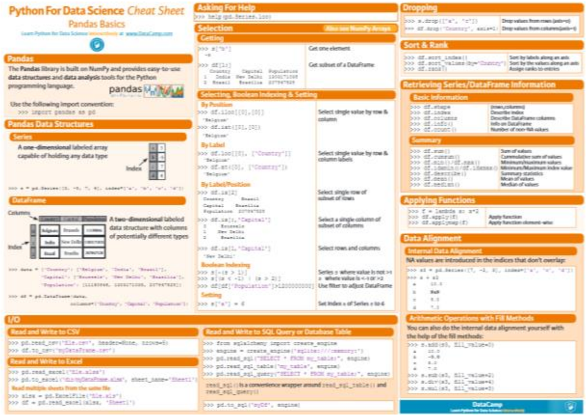
Books at level 3 are intended for young coders that are starting to want to build more involved applications. The first book at this level is *Python: Building Big Apps*. It introduces Object Oriented Programming (OOP) so that readers become familiar with classes and objects which are extensively used in game programming.

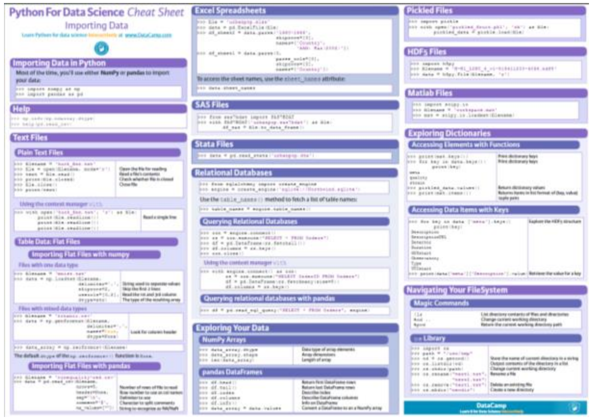
The book does not set out to teach perfect OOP but instead simply aims to introduce some of the concepts and to recommend that building classes allows students to organise their code in big projects in a logical and manageable way.

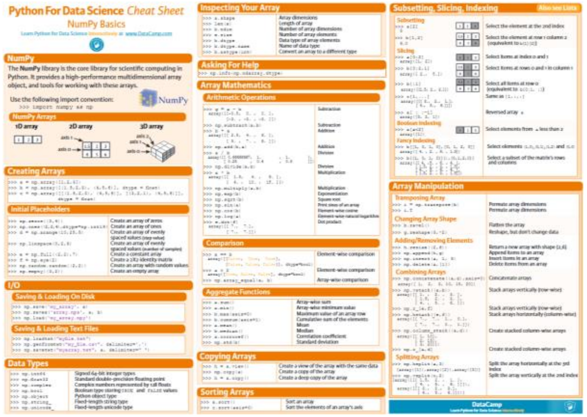
While learning more theory, readers will build a number of small projects ending with a big project - MyPong. The final chapter will show how building MyPong carefully allows us to quickly and easily build a BreakOut style game, an Invaders game and many more.

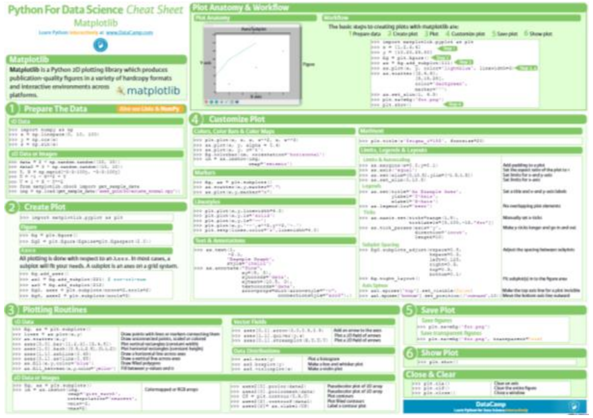
Python Cheat Sheets (DataCamp)

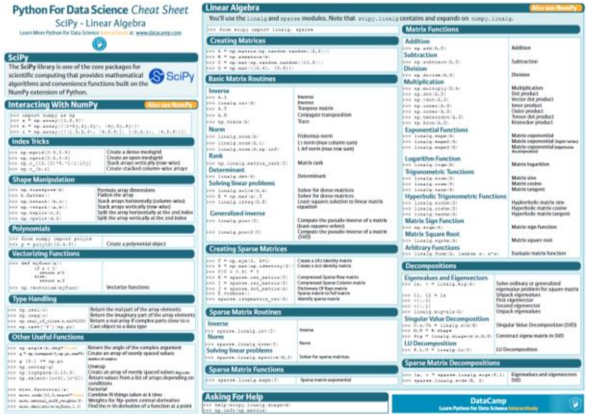

Python For Data Science Cheat Sheet
Python Basics


Python For Data Science Cheat Sheet
Pandas Basics

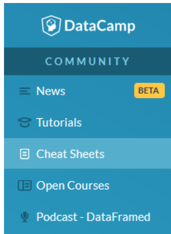

Python For Data Science Cheat Sheet
Importing Data


Python For Data Science Cheat Sheet
NumPy Basics


Python For Data Science Cheat Sheet
Matplotlib


Python For Data Science Cheat Sheet
SciPy - Linear Algebra

… and eight more at DataCamp



Kingston University London

Python Excel Tutorial (DataCamp)

DataCamp

COMMUNITY

- News BETA
- Tutorials
- Cheat Sheets
- Open Courses
- Podcast - DataFramed

DATACAMP

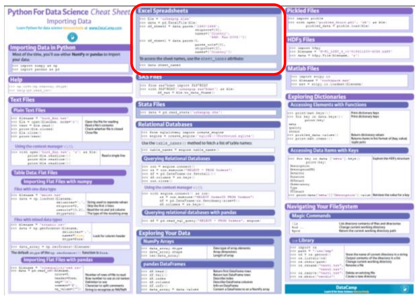
- Official Blog
- Tech Thoughts

Karlijn Willems
January 31st, 2017

MUST READ | IMPORTING & CLEANING DATA | +2

Python Excel Tutorial: The Definitive Guide

Learn how to read and import Excel files in Python, how to write data to these spreadsheets and which are the best packages to do this.

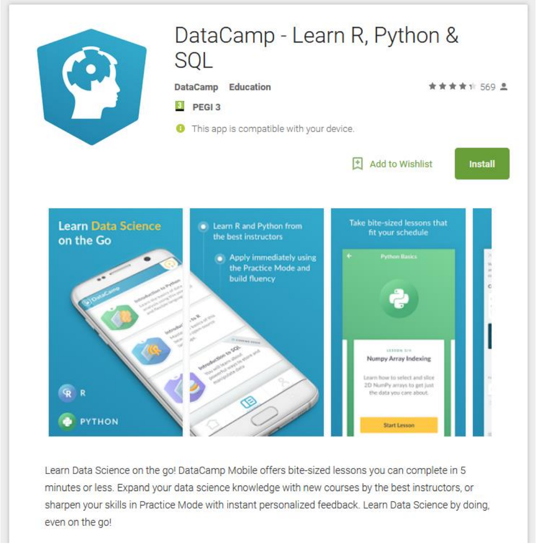


Python For Data Science *Cheat Sheet*
Importing Data

Importing Data in Python | Help | Text Files | Plain Text Files | Excel Spreadsheets | SAS Files | Stata Files | Relational Databases | Pickled Files | HDF5 Files | Matlab Files | Exploring Dictionaries | Accessing Elements with Functions | Accessing Data Items with Keys

Kingston University London

Apps to Learn Python



DataCamp - Learn R, Python & SQL

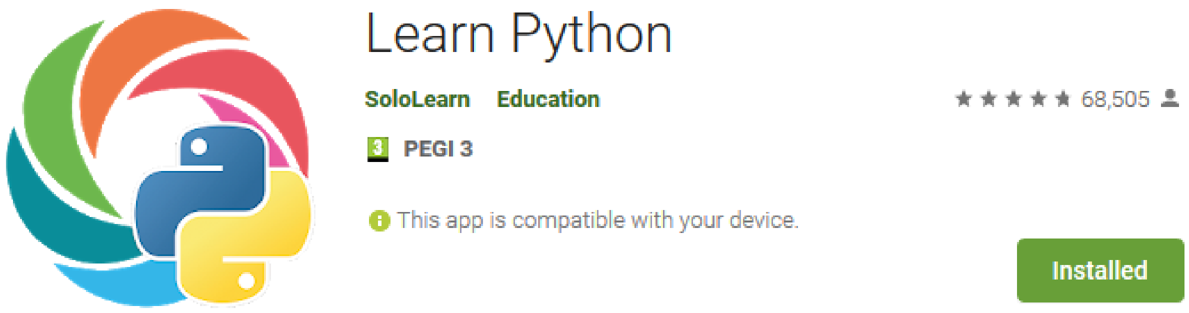
DataCamp Education ★★★★½ 569

PEGI 3

This app is compatible with your device.

Add to Wishlist
Install
Learn Data Science on the Go
R
PYTHON
Learn R and Python from the best instructors
Apply immediately using the Practice Mode and build fluency
Take bite-sized lessons that fit your schedule
Python Basics
Numpy Array Indexing
Start Lesson

Learn Data Science on the go! DataCamp Mobile offers bite-sized lessons you can complete in 5 minutes or less. Expand your data science knowledge with new courses by the best instructors, or sharpen your skills in Practice Mode with instant personalized feedback. Learn Data Science by doing, even on the go!

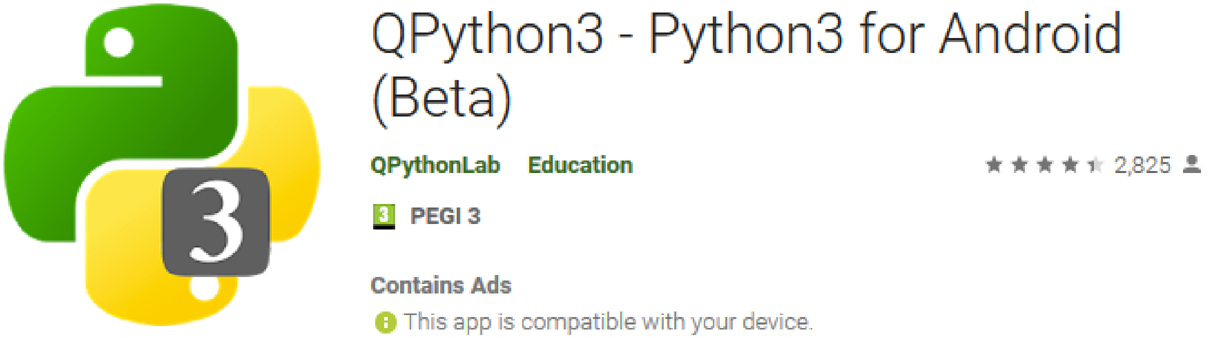


Learn Python

SoloLearn Education ★★★★★ 68,505

PEGI 3

This app is compatible with your device.



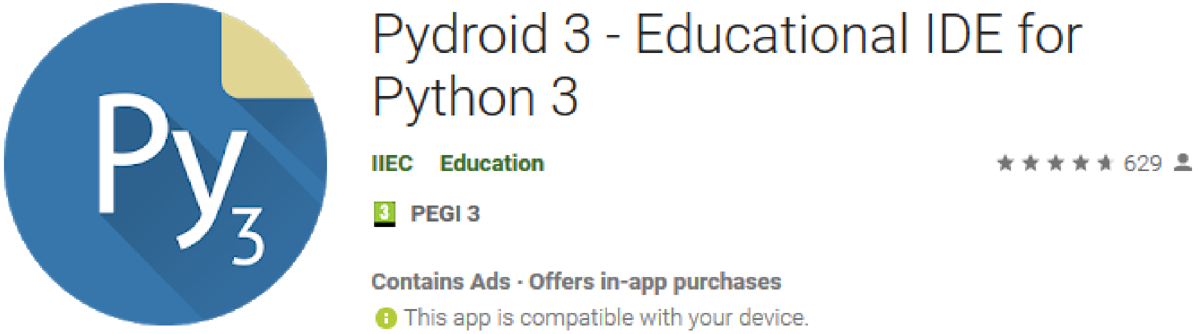
QPython3 - Python3 for Android (Beta)

QPythonLab Education ★★★★½ 2,825

PEGI 3

Contains Ads

This app is compatible with your device.



Pydroid 3 - Educational IDE for Python 3

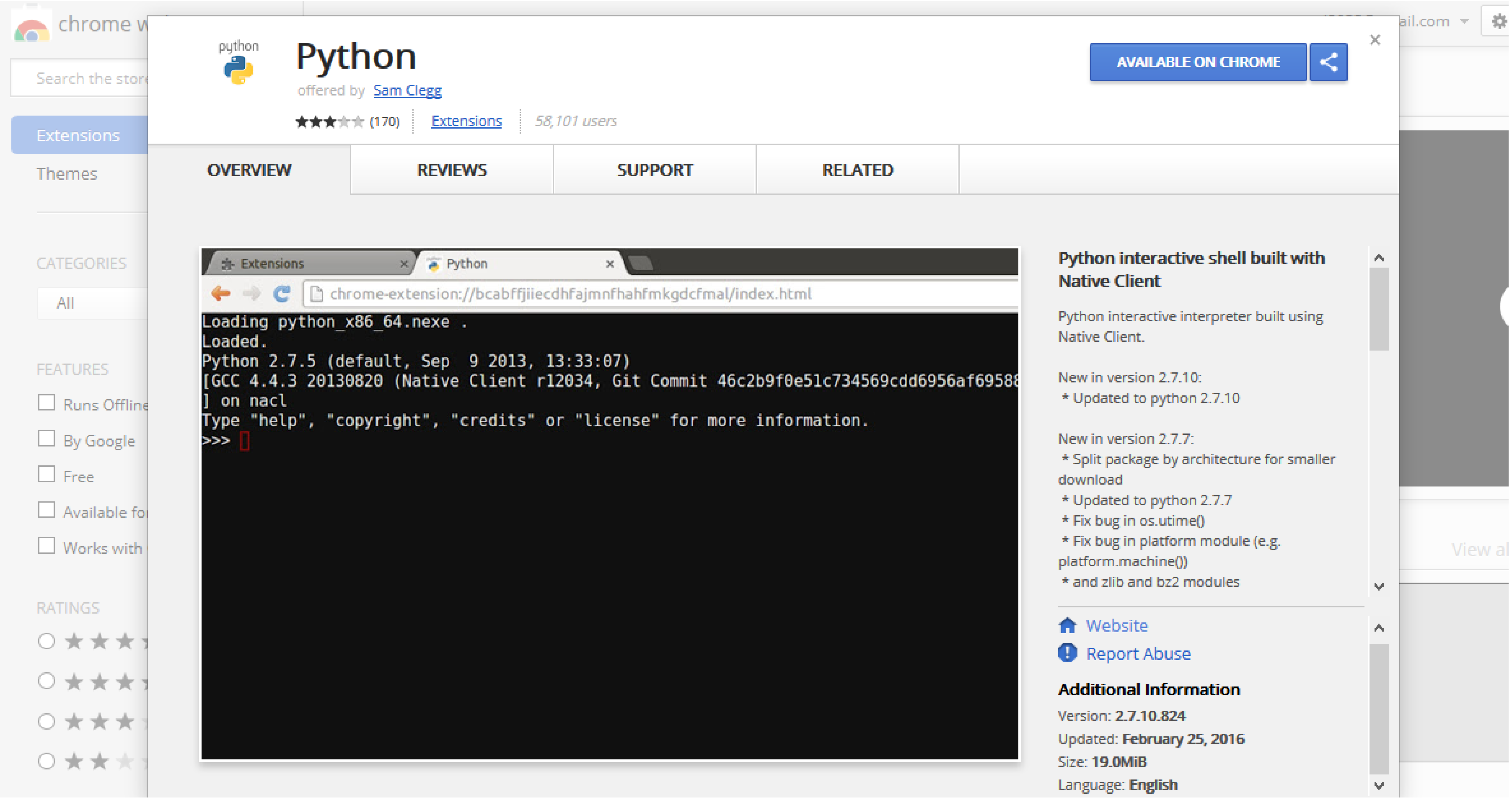
IIEC Education ★★★★★ 629

PEGI 3

Contains Ads · Offers in-app purchases

This app is compatible with your device.

Chrome Extension (Python)



Python

offered by Sam Clegg

★★★☆☆ (170) | Extensions | 58,101 users

```
Loading python_x86_64.nexe .
Loaded.
Python 2.7.5 (default, Sep  9 2013, 13:33:07)
[GCC 4.4.3 20130820 (Native Client r12034, Git Commit 46c2b9f0e51c734569cdd6956af6958
] on nacl
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Python interactive shell built with Native Client

Python interactive interpreter built using Native Client.

New in version 2.7.10:
 * Updated to python 2.7.10

New in version 2.7.7:
 * Split package by architecture for smaller download
 * Updated to python 2.7.7
 * Fix bug in os.utime()
 * Fix bug in platform module (e.g. platform.machine())
 * and zlib and bz2 modules

Website

Report Abuse

Additional Information

Version: **2.7.10.824**
Updated: **February 25, 2016**
Size: **19.0MiB**
Language: **English**

… Looking forward

Day 2
Introduction to Python as a coding language for bioinformatics
What can you do with programming languages?
What is Python and why Python?
Python basics
Simple Python codes
Demonstrations of more advanced methods
Questions & answers