Set-up Python

Get Python setup on your own computer

Background:

The Codecademy site provides you with a development environment already set up for you to complete exercises and run your code. Now that you've learned how to code in Python, let's go through the process of setting up your development environment on your computer so that you can write your own applications.

While you took the Python Codecademy course, you've been submitting code to us which we have interpreted, run, and then sent the results back to you. This article will walk you through how to set up Python and execute code on your own computer. Sounds intimidating, but there are some nice tools to help you out that we'll introduce you to. You'll be flying in no time.

Why build outside of Codecademy? The world of programming is massive, and it's impossible to teach everything in one place. While Codecademy is excellent for teaching lessons, it also has limitations. For instance, it's difficult to create large projects and share them with the world using only the tools found in Codecademy's Python course. This guide will teach you:

1. How to download and install python
2. How to run python from the command line
3. How to install python libraries using pip, a popular package manager for python (we'll explain what that means!)

Downloading Python

When you learned using the Codecademy course, you wrote the solutions in your web browser and submitted the data to us so you did not need to have Python installed on your computer. Now, you're going to write and run actual programs on your computer.

Visit the official Python downloads page and find the most recent release of Python 2.7 that corresponds to your OS (operating system).

Make sure to download Python 2.x and not Python 3.x. Although Python 3 has a higher version number, it isn't completely compatible with Python 2, the version you learned in the Codecademy course. Both versions are widely used and actively updated.

After your download has completed, launch the installer. This may require that you unzip the file first depending on the version of the installer you downloaded. Follow the instructions provided by the installer to complete your Python installation.

What did I just install?

Computers read and execute code based on machine language which is stored in hexadecimal format. It is virtually impossible for a human to write in machine language and each processor has its own version.

To make programming simpler, human-readable languages like Python were invented. The files you just installed include a Python interpreter. This interpreter converts your human-readable Python code into instructions that the computer can act on.

Running Python:

Once you have downloaded Python, you should be able to pull up your computer's terminal and start running it. On Windows, search for a program called "cmd" and then launch it If you're on a Mac or a Linux environment look for and launch the program "terminal." You should have a command-line prompt that looks similar to this:

```
Adam@ADAM-PC ~
$
```

If you downloaded Python properly, you can just type `python` into the prompt and hit enter to get a result like the below:

```
Adam@ADAM-PC ~
$ python
Python 2.7.10 (v2.7.10:648dcafa7e5f, Dec 10 2014, 10:10:46)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

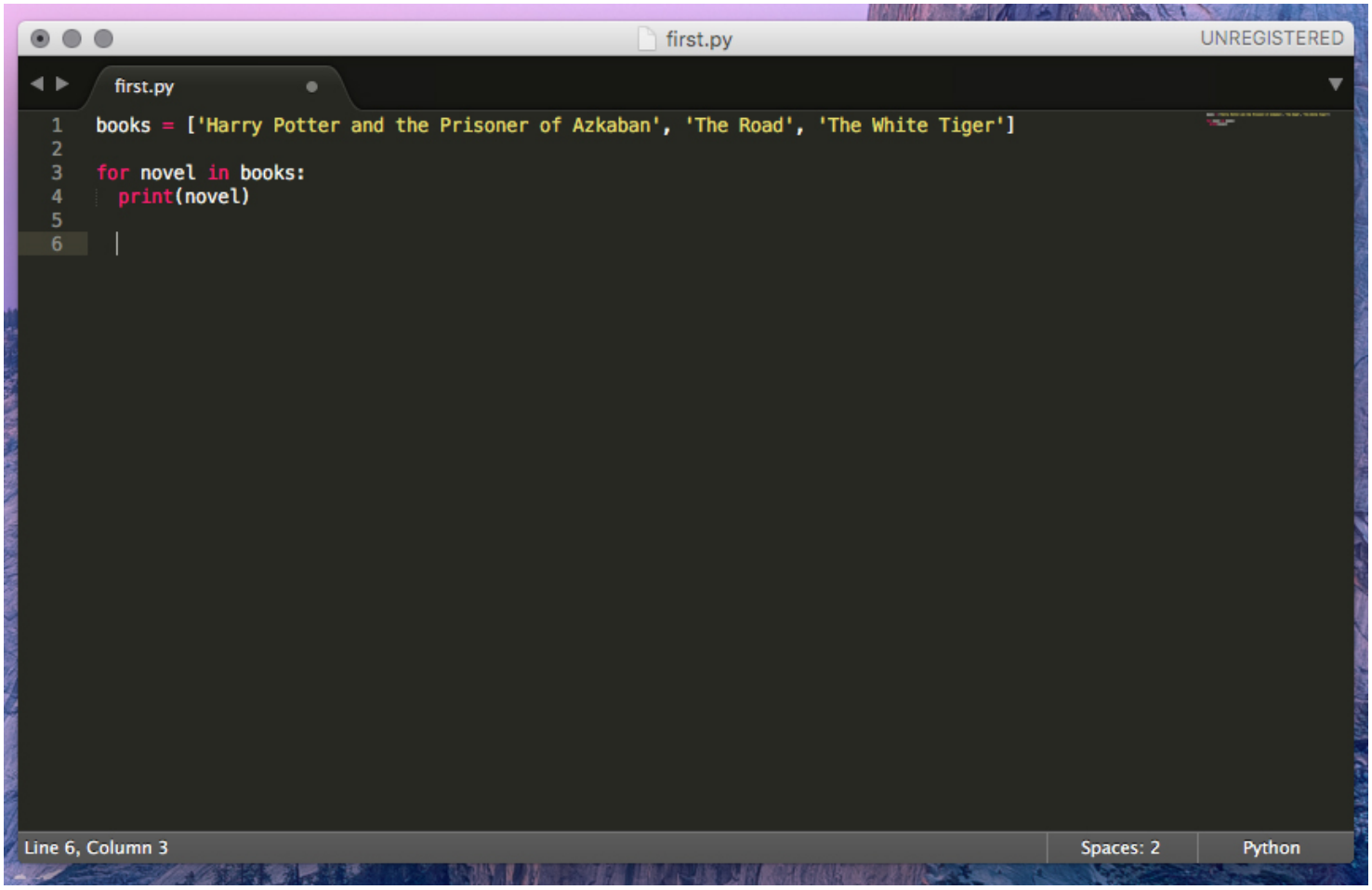
From here we can type in Python code and watch it be interpreted for us on the fly. Go ahead and type some simple stuff in and watch Python work right in front of you:

```
Adam@ADAM-PC ~
$ python
Python 2.7.10 (v2.7.10:648dcafa7e5f, Dec 10 2014, 10:10:46)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> movies = ["Birdman", "Black Swan", "Les Mis"]
>>> for film in movies:
...     print(film)
...
Birdman
Black Sawn
Les Mis
>>> 5 + 5
10
>>>
```

The command line is useful for checking simple code, but if you're looking to build something more involved, you'll benefit by starting with a text editor. Let's exit out. We'll make our own files and return to the command line to run those.

If you do not have a good text editor for editing code, we recommend Sublime Text, for which we have a step-by-step setup guide here(link). Any text editor will work but may require some additional setup.

Once you're set-up, write a simple piece of Python code like the below in your text editor. Save it, making sure the file name ends with a .py extension. We'll name our file first.py for this example.



Now let's run your code and see what we get. Pull up your terminal again and locate the directory in which you saved your python file. Use the cd command to get there. (If you're rusty on how to locate and change directories, refer to our Learn the Command Line course). Then type `python first.py`. Your terminal window should look something like this:

```
Adam@ADAM-PC ~
$ python first.py
Harry Potter
The Road
The White Tiger
```

The output of your script should print in your terminal window. If you're seeing errors or nothing at all, make sure you're in the same directory as your code, and that your code has a print statement that gets executed. (In other words, that you're telling your code to actually print something.)

If it printed, congratulations! You've now written and run a Python file all on your own! Now you're equipped to create those projects that you've dreamed of.

Making big projects takes lots of work. Lots and lots of work, so much so that it's useful to have a little help from others. Thanks to the very open culture of programming, there are many open source libraries out there for you to use. Using libraries help us work quickly, and allow us not to reinvent the wheel every time we sit down to code.

Using PIP to get and install Packages:

As you begin to build your own applications with Python, you'll likely use a lot of packages. The best way to install and manage them is with a package manager.The most popular and most recommended package manager for Python is called pip.

Pip should already be included with your installation of Python. To check, just type `pip` into your command line window and see what pops up. If you get something like below, then you have pip installed. If you get an error, install the latest version of pip here.

```
Adam@ADAM-PC ~
$ pip

Usage:
  pip <command> [options]

Commands:
  install                     Install packages.
  uninstall                   Uninstall packages.
  freeze                      Output installed packages in requirements format.
  list                        List installed packages.
  show                        Show information about installed packages.
  search                      Search PyPI for packages.
  wheel                       Build wheels from your requirements.
  zip                         DEPRECATED. Zip individual packages.
  unzip                       DEPRECATED. Unzip individual packages.
  help                        Show help for commands.

General Options:
  -h, --help                  Show help.
  --isolated                  Run pip in an isolated mode, ignoring environment
                              variables and user configuration.
  -v, --verbose               Give more output. Option is additive, and can be
                              used up to 3 times.
  -V, --version               Show version and exit.
  -q, --quiet                 Give less output.
  --log <path>                Path to a verbose appending log.
  --proxy <proxy>             Specify a proxy in the form
                              [user:passwd@]proxy.server:port.
  --retries <retries>         Maximum number of retries each connection should
                              attempt (default 5 times).
  --timeout <sec>             Set the socket timeout (default 15 seconds).
  --exists-action <action>    Default action when a path already exists:
                              (s)witch, (i)gnore, (w)ipe, (b)ackup.
  --trusted-host <hostname>   Mark this host as trusted, even though it does
                              not have valid or any HTTPS.
  --cert <path>               Path to alternate CA bundle.
  --client-cert <path>        Path to SSL client certificate, a single file
                              containing the private key and the certificate
                              in PEM format.
  --cache-dir <dir>           Store the cache data in <dir>.
  --no-cache-dir              Disable the cache.
  --disable-pip-version-check
                              Don't periodically check PyPI to determine
                              whether a new version of pip is available for
                              download. Implied with --no-index.
```

How does pip work?

To download and install packages, pip searches through PyPi, the Python Package Index, a large repository of registered open source python packages.

```
Adam@ADAM-PC /d/Codecademy/Python Articles/Set-Up/Code
$ pip list
mysql-connector-python (2.1.2)
nltk (3.0.5)
pip (1.5.4)
queuelib (1.4.2)
Scrapy (1.0.3)
setuptools (2.1)
six (1.9.0)
Twisted (15.4.0)
w3lib (1.12.0)
XlsxWriter (0.7.3)

Adam@ADAM-PC /d/Codecademy/Python Articles/Set-Up/Code
$
```

Let's try installing a package called scrapy. It's used to gather data from websites by "scraping" information from them. More info on Scrapy can be found on their official site. To install Scrapy, go to your terminal and enter `pip install scrapy`. In my terminal below, I also used "pip list" to show all my installed packages before I installed scrapy and then after it was installed to verify that I successfully installed it.

From now on, we can use scrapy in any of our Python applications. In order to use it, type "import scrapy" in the python file that makes calls to it.

```
Adam@ADAM-PC /d/Codecademy/Python Articles/Set-Up/Code
$ pip list
mysql-connector-python (2.1.2)
nltk (3.0.5)
pip (1.5.4)
setuptools (2.1)
six (1.9.0)
XlsxWriter (0.7.3)

Adam@ADAM-PC /d/Codecademy/Python Articles/Set-Up/Code
$ pip install scrapy
Downloading/unpacking scrapy
  Running setup.py (path:C:\Users\Adam\AppData\Local\Temp\pip_build_Adam\scrapy\setup.py) egg_info for package scrapy

    no previously-included directories found matching 'docs\build'
    warning: no files found matching '*' under directory 'bin'
Downloading/unpacking Twisted>=10.0.0 (from scrapy)
  Running setup.py (path:C:\Users\Adam\AppData\Local\Temp\pip_build_Adam\Twisted\setup.py) egg_info for package Twisted

    file twisted.py (for module twisted) not found
    file twisted\application.py (for module twisted.application) not found
    file twisted\application\test.py (for module twisted.application.test) not found
    file twisted\cred.py (for module twisted.cred) not found
    file twisted\cred\test.py (for module twisted.cred.test) not found
    file twisted\internet.py (for module twisted.internet) not found
    file twisted\internet\test.py (for module twisted.internet.test) not found
    file twisted\logger.py (for module twisted.logger) not found
    file twisted\logger\test.py (for module twisted.logger.test) not found
    file twisted\names.py (for module twisted.names) not found
    file twisted\names\test.py (for module twisted.names.test) not found
    file twisted\persisted.py (for module twisted.persisted) not found
    file twisted\plugins.py (for module twisted.plugins) not found
    file twisted\positioning.py (for module twisted.positioning) not found
    file twisted\protocols.py (for module twisted.protocols) not found
    file twisted\protocols\test.py (for module twisted.protocols.test) not found
    file twisted\python.py (for module twisted.python) not found
    file twisted\python\test.py (for module twisted.python.test) not found
    file twisted\scripts.py (for module twisted.scripts) not found
    file twisted\test.py (for module twisted.test) not found
```

If you decide you want to uninstall scrapy, just use the command: `pip uninstall scrapy`.

```
Adam@ADAM-PC /d/Codecademy/Python Articles/Set-Up/Code
$ pip list
mysql-connector-python (2.1.2)
nltk (3.0.5)
pip (1.5.4)
queuelib (1.4.2)
Scrapy (1.0.3)
setuptools (2.1)
six (1.9.0)
Twisted (15.4.0)
w3lib (1.12.0)
XlsxWriter (0.7.3)

Adam@ADAM-PC /d/Codecademy/Python Articles/Set-Up/Code
$ pip uninstall scrapy
Uninstalling Scrapy:
  c:\python34\lib\site-packages\scrapy-1.0.3-py3.4.egg-info
  c:\python34\lib\site-packages\scrapy\__init__.py
  c:\python34\lib\site-packages\scrapy\__pycache__\__init__.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\_monkeypatches.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\cmdline.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\command.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\conf.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\crawler.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\dupefilter.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\dupefilters.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\exceptions.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\exporters.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\extension.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\interfaces.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\item.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\link.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\linkextractor.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\log.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\logformatter.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\mail.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\middleware.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\project.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\resolver.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\__pycache__\responsetypes.cpython-34.pyc
```

Use `pip list` again to make sure you uninstalled properly.

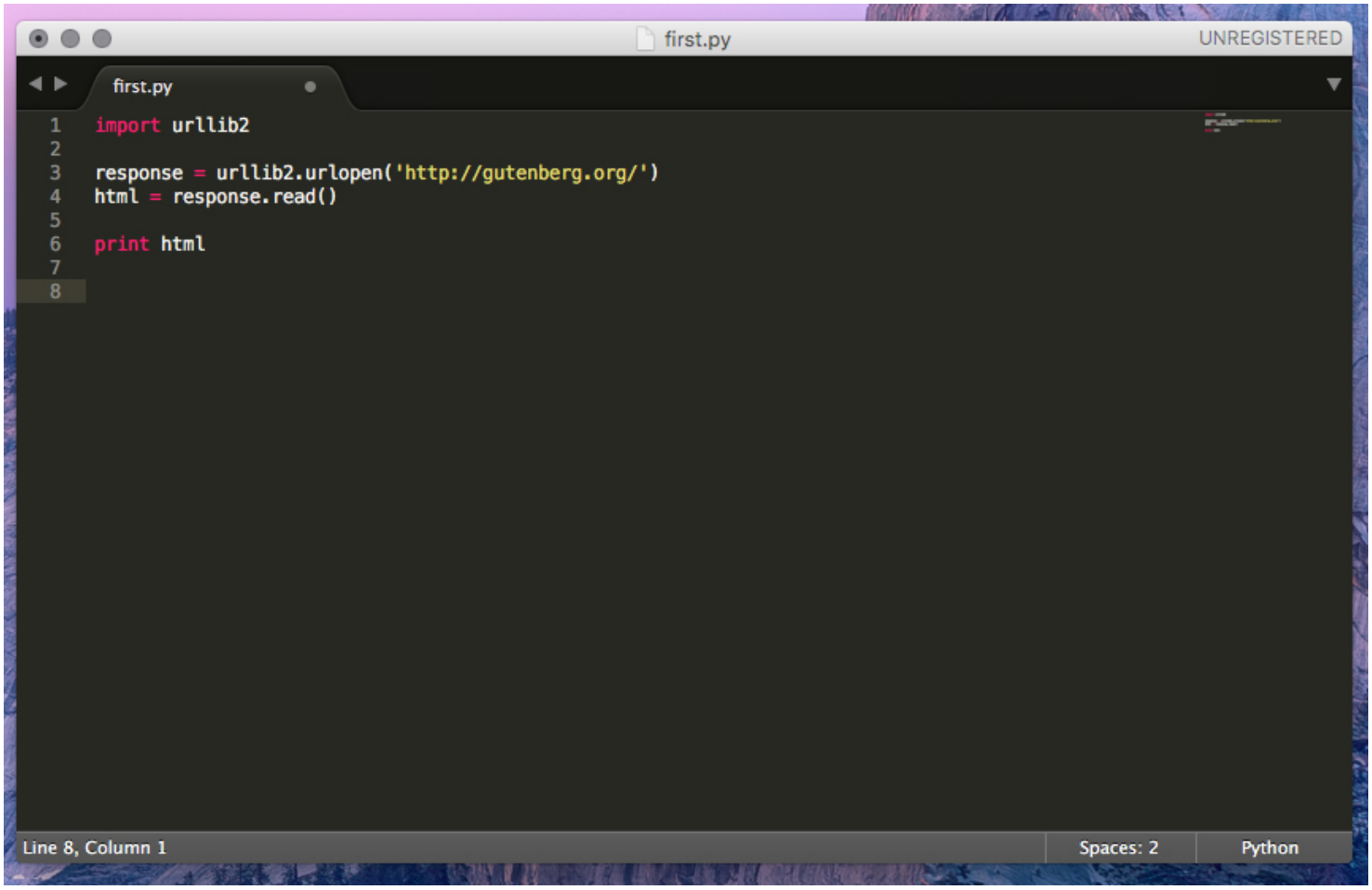
```
  c:\python34\lib\site-packages\scrapy\xlib\pydispatch\__pycache__\robust.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\xlib\pydispatch\__pycache__\robustapply.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\xlib\pydispatch\__pycache__\saferef.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\xlib\pydispatch\dispatcher.py
  c:\python34\lib\site-packages\scrapy\xlib\pydispatch\errors.py
  c:\python34\lib\site-packages\scrapy\xlib\pydispatch\license.txt
  c:\python34\lib\site-packages\scrapy\xlib\pydispatch\robust.py
  c:\python34\lib\site-packages\scrapy\xlib\pydispatch\robustapply.py
  c:\python34\lib\site-packages\scrapy\xlib\pydispatch\saferef.py
  c:\python34\lib\site-packages\scrapy\xlib\tx\__init__.py
  c:\python34\lib\site-packages\scrapy\xlib\tx\__pycache__\__init__.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\xlib\tx\__pycache__\_newclient.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\xlib\tx\__pycache__\client.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\xlib\tx\__pycache__\endpoints.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\xlib\tx\__pycache__\interfaces.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\xlib\tx\__pycache__\iweb.cpython-34.pyc
  c:\python34\lib\site-packages\scrapy\xlib\tx\_newclient.py
  c:\python34\lib\site-packages\scrapy\xlib\tx\client.py
  c:\python34\lib\site-packages\scrapy\xlib\tx\endpoints.py
  c:\python34\lib\site-packages\scrapy\xlib\tx\interfaces.py
  c:\python34\lib\site-packages\scrapy\xlib\tx\iweb.py
  c:\python34\scripts\scrapy-script.py
  c:\python34\scripts\scrapy.exe
Proceed (y/n)? y
  Successfully uninstalled Scrapy

Adam@ADAM-PC /d/Codecademy/Python Articles/Set-Up/Code
$ pip list
mysql-connector-python (2.1.2)
nltk (3.0.5)
pip (1.5.4)
queuelib (1.4.2)
setuptools (2.1)
six (1.9.0)
Twisted (15.4.0)
w3lib (1.12.0)
XlsxWriter (0.7.3)

Adam@ADAM-PC /d/Codecademy/Python Articles/Set-Up/Code
$
```

Pro Tip: If you're working on something and wondering if there's an easier way to do something, there's probably already an open sourced a package that accomplishes the task. You can use a search engine or search through PyPi itself to find what you want. Can't find what you want? Consider writing your own package and releasing it as open source so others can use it.

Retrieve Information from the Web:

There are many different packages that can fetch data from the web including scrapy. Let's go through an example using one of Python's built-in packages called urllib2

Since urllib2 is built into Python, we do not need to install it using pip. Let's use it to grab data from Project Gutenberg, a massive collection of free eBooks. Follow the code shown below, save it and run it.



Woah! That's a lot of HTML!

Let's look at what we did.

First we requested the url at Project Gutenberg's homepage which sent a response that we printed. If we had made the same request with a web browser, the browser would have interpreted the HTML for us and displayed a pretty picture. Instead, our application read the raw HTML file for us and printed the output. This may look messy, but it can be useful for looking through websites in order to get information from them.

Let's say we wanted to count how many times an author uses a word in a book. Let's use one of my favorites, Les Miserables by Victor Hugo. First, we find the url for the book on Project Gutenberg, open the url, read it, and then parse it to see how many times Hugo uses the word "sad."

```
import urllib2

response = urllib2.urlopen('http://gutenberg.org/')
html = response.read()

sad = 0

list_of_words = html.split(' ')

for word in list_of_words:
  if word == 'sad':
    sad += 1

print sad
```

Looks like this is a slightly depressing book! This is just one basic usage example of urllib2 and web scrapers in general, and is meant to give you a taste of what's possible when you use Python locally.

If you're interested, a more in-depth tutorial on urllib2 can be found in Python's official documentation. You can refer to Python's documentation to browse all the features of a particular package in the Python standard library.

Teaching the world how to code.

Company

- About
- Stories
- We're hiring
- Blog

Resources

- Articles
- Schools

Learn To Code

- Make a Website
- Make an Interactive Website
- Learn Rails
- Ruby on Rails Authentication