

PyLadies API Workshop - Installation

Installation Checklist:

- Computer Setup
 - Mac 10.7 and newer:
 - ☐ zlib, libpng, & freetype installation
 - ☐ pip installation
 - ☐ virtualenv installation
 - ☐ Python Package installation
 - ☐ Sanity Check - are all Python packages installed?
 - ☐ Installed a text editor (if you don't have one already)
 - Linux:
 - ☐ update & upgrade system
 - ☐ install compiler, libjpeg, libpng, libfreetype, python dependencies
 - ☐ install virtualenv
 - ☐ Python Package installation
 - ☐ Sanity Check - are all Python packages installed?
 - ☐ Installed a text editor (if you don't have one already)
 - Windows, Mac 10.6 and older, and anyone else having issues:
 - ☐ PythonAnywhere
 - ☐ Python Package installation
 - ☐ Sanity Check - are all Python packages installed?
 - ☐ Installed a text editor (if you don't have one already)
- Application Setup
 - ☐ EchoNest
 - ☐ Spotify
 - ☐ GitHub
- ☐ Before you leave!

— Computer Setup —

NOTE!!

For Mac or Linux machine having trouble, please feel free to do the Windows/older version of Mac OSX

setup as it is all online!

Have a Mac and don't know what version of OS you have?

1. Click on the Apple icon in the upper left corner, and select "About this Mac".
2. A little window will pop up where you can find your version:



Your Mac OS Version

Mac OS X 10.7 and newer (Lion, Mountain Lion & Mavericks)

NOTE: To open up your Terminal application (also called Console):

1. Open up Finder
2. Navigate to "Applications"
3. Navigate to "Utilities"
4. Double-click to launch "Terminal"

I. Command Line Tools

If you don't already have one, you'll need to create a (free) developer account (attached to your Apple iTunes login). Go [here](#) to complete the registration.

After you register, navigate [here](#) and find "Command Line Tools" for your version of your Mac. Download & install the package.

To verify that it is all correct, go to your Terminal, and type: `gcc`. You *should* see something like the following:

```
clang: error: no input files
```

If you see `command not found: gcc`, please grab a mentor and ask for help.

II. ZLIB

Have Homebrew installed? If you have homebrew installed (no need to install it!), run first run `brew update`, then `brew install zlib`. Continue onto “II. LIBPNG”.

Don’t have Homebrew installed? Go to <http://www.zlib.net/> and download the latest source code archive. Scroll down to zlib source code, and download version 1.2.7, tar.gz format

In your Downloads folder, double-click the zip file to “unpack” it. Next, in your Terminal window, type the following (no need to type the `$!`):

```
$ cd ~/Downloads
$ cd zlib-1.2.7
$ ./configure
$ make
$ sudo make install
```

GOT AN ERROR? Ask for help!

III. LIBPNG

Have Homebrew installed? If you have homebrew installed (no need to install it!), run `brew install libpng`. Continue onto “III. FREETYPE”.

Don’t have Homebrew installed? Go to <http://www.libpng.org/pub/png/libpng.html> and download the latest source code archive. Scroll down and download by clicking the “tar.gz” link on the download.sourceforge.net row.

In your Downloads folder, double-click the zip file to “unpack” it. Next, in your Terminal window, type the following (no need to type the `$!`):

```
$ cd ~/Downloads
$ cd libpng-1.5.14
$ ./configure
$ make
$ make check
$ sudo make install
```

GOT AN ERROR? Ask for help!

IV. FREETYPE

Have Homebrew installed? If you have homebrew installed (no need to install it!), run `brew install freetype`. Continue onto “IV. Python Setup”.

Don’t have Homebrew installed? Go to <http://sourceforge.net/projects/freetype/files/latest/download?source=files> and the download of the source file should start automatically

In your Downloads folder, double-click the zip file to “unpack” it. Next, in your Terminal window, type the following (no need to type the `$!`):

```
$ cd ~/Downloads
$ cd freetype-2.4.11
$ ./configure
$ make
$ sudo make install
```

GOT AN ERROR? Ask for help!

V. Python setup

1. Install pip by following instructions here: [get-pip](#).
2. After you install `pip` with the previous step, run the following commands in your terminal (if prompted for your password, use your password that you use to login to your computer):

```
$ sudo pip install virtualenv
```

GOT AN ERROR? Ask for help!

VI. Python Package Installation

For Python Package installation, continue on to **Everyone** below.

Linux (Ubuntu/Debian)

In a terminal:

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install build-essential python-dev python-pip libjpeg8-dev libfreetype6-dev libpng12-dev
$ sudo pip install virtualenv
```

GOT AN ERROR? Ask for help!

Next, continue on to **Everyone** below.

Linux (Fedora/RHEL)

Note: If `yum <command> <package>` throws an error (it will be my fault because I am trying to do this from memory), I probably gave you the incorrect package name or command. Google “fedora” where package name is the name minus the ‘devel’, e.g. “fedora install libpng”. My apologies for any error!

In a terminal

```
$ sudo yum update
$ sudo yum upgrade
$ sudo yum groupinstall "Development Tools"
$ sudo yum install python-devel python-pip libpng-devel freetype-devel libjpeg-devel
$ sudo pip install virtualenv
```

GOT AN ERROR? Ask for help!

Next, continue on to **Everyone** below.

Windows and Mac OS X 10.6 and older

And for anyone with issues setting up their machine :-)

1. Make a free account on [PythonAnywhere](#).
2. Once your account is made, login, and click on “Bash” under “Start a New Console”:

Start a new console:Python: [2.7](#) / [2.6](#) / [3.3](#) / [3.4](#) IPython: [2.7](#) / [2.6](#) / [3.3](#) / [3.4](#) PyPy: [2.7](#)Other: [Bash](#) | [MySQL](#)*PythonAnywhere: Bash console*Continue on to **Everyone** below.

Everyone

Python library installation

Within the console/terminal, setup your project directory with the following commands (no need to type the `$!`):

```
~ $ cd
~ $ mkdir api-project
~ $ cd api-project
~/api-project $ virtualenv env
~/api-project $ source env/bin/activate
# if all is good, you should now see the (env):
(env) ~/api-project $:
```

GOT AN ERROR? Ask for help!

Now for installing:

```
(env) ~/api-project $ pip install github3.py geojson geopy pyen numpy requests
(env) ~/api-project $ pip install matplotlib # for some reason only works *after* numpy is installed
(env) ~/api-project $
```

GOT AN ERROR? Ask for help!

For Mac OSX Mavericks:

For matplotlib to work on Mavericks, *copy and paste* the following commands into your terminal:

First, copy & paste:

```
mkdir ~/.matplotlib
```

Then, copy & paste:

```
echo "backend: TkAgg" >> ~/.matplotlib/matplotlibrc
```

GOT AN ERROR? Ask for help!

Check your setup

1. Close your terminal, and reopen it - let's start fresh!
2. In your terminal, type:

```
$ cd
```

This gets into your “Home” directory. `cd` means “change directory”, and we had nothing after it, so it defaults to “Home”. You may see “Home” denoted with a `~` sometimes - it means the same.

3. Now move into your project directory by using the same “change directory” command, followed by the new project directory name we created:

```
$ cd api-project
```

Now, if no errors happened, you may (or may not, or something similar, depending on how your console is setup) see the directory name preceding the `$`:

```
~/api-project $
```

GOT AN ERROR? Ask for help!

4. Now to (re-)activate the virtualenv for the project:

```
~/api-project $ source env/bin/activate
```

Here, we “source”-ed a file called “activate”. “source” literally executes the contents of the file after it. Here that file is named “activate” within the `env/bin/` directory.

REMEMBER!: Remember the above command, `source env/bin/activate` to turn “on” your virtualenv. If all is okay, you should see:

```
(env) ~/api-project $
```

GOT AN ERROR? Or don’t see the `(env)`? Ask for help!

5. Let’s quickly get familiar with activating & deactivating our virtualenv. Type the following to turn “off” the virtualenv named `(env)`:

```
(env) ~/api-project $ deactivate
```

You should see your Terminal prompt return to normal, without the `(env)` preceding your directory location and `$`:

```
~/api-project $
```

Now let’s reactivate! Do you remember what to do? Hint: it’s that `source env/bin/activate` command!

REMEMBER!: Remember the above commands, `source env/bin/activate` to turn “on” your virtualenv, and `deactivate` to turn it off when you are done for the day.

6. Next let’s run a Python shell:

```
(env) api-project $ python
```

And you should see something like this:

```
Python 2.7.8 (default, Sep 12 2014, 10:03:22)
[GCC 4.2.1 Compatible Apple LLVM 5.1 (clang-503.0.40)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

GOT AN ERROR? Ask for help!

7. Now, let’s see if we can literally import all the Python packages that we `pip installed` earlier. Type the following to see if we can correctly import `geojson`:


```
>>> import geojson
```

With each `import` statement, **you should not see anything happen**, just another `>>>` prompt. That means they have been installed just fine.

However, if you see any error message like the following, please ask for help:

```
>>> import foo
Traceback (most recent call last):
  File "<stdin>", line 1, in
ImportError: No module named foo
```

8. Now continue on testing with the rest of the packages:

```
>>> import geopy
>>> import pyen
>>> import numpy
>>> import matplotlib
>>> import github3
>>> import requests
```

Python setup is complete! Please continue below with “Install a Text Editor”.

Install a Text Editor

If you do not have a text editor already (e.g. Sublime Text 2 or 3, TextMate, Notepad++, gedit, Vim, Emacs, etc. **NOT** TextEdit, Word, or any other word processor), I’d suggest installing Sublime Text 2. It’s free (it will just bug you to register every 10 times you save or so, just ignore).

Sublime Text 2 can be downloaded [here](#).

We need a text editor to actually write our Python code.

All done? Once you have a text editor, continue to “App Registration/Setup” down below.

NOTE FOR PYTHONANYWHERE FOLKS

With PythonAnywhere, you can either:

- write code PythonAnywhere instead of in a text editor like Sublime Text:
 1. Click on “Files”:

Start a new console:

Python: [2.7](#) / [2.6](#) / [3.3](#) / [3.4](#) | IPython: [2.7](#) / [2.6](#) / [3.3](#) / [3.4](#) | PyPy: [2.7](#)
Other: [Bash](#) | [MySQL](#)

Your consoles:

[Bash console 793375 - kill](#)

Consoles shared with you

No-one has shared any consoles with you :-(

- Click on “api-project” directory - the directory you made above:

`/ > home >`  **roguelynn**

Enter new directory name

New

 [.cache/](#)



 [.config/](#)



 [.pip/](#)



 [.virtualenvs/](#)



 [Dropbox/](#)

 [api-project/](#)



 [new-coder/](#)



- (when we are at the workshop Saturday) Create a new file via:

`/ > home > roguelynn >`  **api-project**

 [Open Bash console here](#)

Enter new directory name

New

 [env/](#)



Enter new file name

New

Upload a file: [Choose File](#) No file chosen

- or write code on Sublime Text or any other text editor and upload it to PythonAnywhere to run:

- Click on “Files”:

Start a new console:

Python: [2.7](#) / [2.6](#) / [3.3](#) / [3.4](#) | IPython: [2.7](#) / [2.6](#) / [3.3](#) / [3.4](#) | PyPy: [2.7](#)
 Other: [Bash](#) | [MySQL](#)

Your consoles:

[Bash console 793375 - kill](#)

Consoles shared with you

No-one has shared any consoles with you :-{

- Click on “api-project” directory - the directory you made above:

`/ > home >`  **roguelynn**

 [.cache/](#)



 [.config/](#)



 [.pip/](#)



 [.virtualenvs/](#)



 [Dropbox/](#)

 **[api-project/](#)**



 [new-coder/](#)



- (when we are at the workshop Saturday) Click on “Upload a file” and select the Python file to upload:

`/ > home > roguelynn >`  **api-project**

 [Open Bash console here](#)

 [env/](#)



Upload a file: No file chosen

When I say “within your terminal” or “console”, this is in reference to running code. The easiest way for PythonAnywhere folks to get to the terminal/console after editing or uploading a file, click “Open Bash Console Here”:

 [env/](#)  Upload a file: No file chosen

Another option is going back to “Consoles”, clicking the one console under “Your Console”. You should see something like this:

```
(env) roguelynn@giles-liveconsole2:~/api-project$
```

Notice at the end, the `~/api-project$`. This is telling me I am in the `api-project` directory, and this is good!

If you are not taken to where you were left off (e.g. in the same directory, `api-project`, with the `(env)` activated, type the following:

```
roguelynn@giles-liveconsole2:~$ cd api-project
roguelynn@giles-liveconsole2:~/api-project$ source env/bin/activate
(env)roguelynn@giles-liveconsole2:~/api-project$
```

Feel free to play around here to get familiar with going back and forth from your Files and Console, making sure the `(env)` is activated.

— App Registration/Setup —

In working with a company’s APIs, you often need to register with their website so you can get a “key” to access the API. Registration is often free for small-time devs like us.

We will need to register for EchoNest and Spotify to use their APIs. If you have a GitHub account, please follow the GitHub instructions. Otherwise, there’s no need to create a GitHub account for now.

Create a config file

First, in your `api-project` directory, create a file called `config.ini`. We will save all of our developer keys

from EchoNest, Spotify, and GitHub here. Copy and paste the following text into your `config.ini` file:

[echonest]

api_key =

[spotify]

client_id =

client_secret =

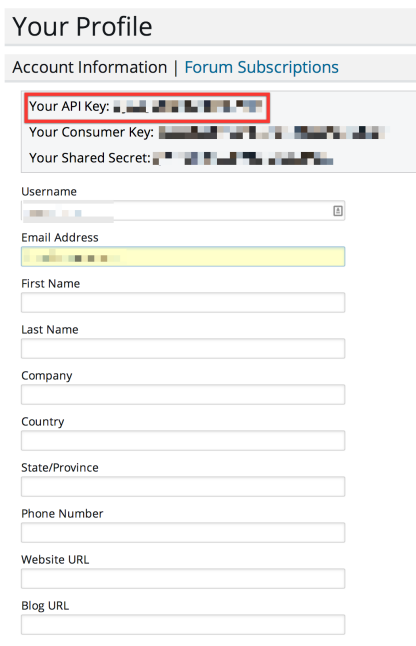
[github]

oauth =

EchoNest

Developer's site: [EchoNest](#)

1. If you don't already, create a free developer's account for EchoNest [here](#). Be sure to verify your account through the email they send.
2. Once your account is created, you should have the API key information already available in your [account profile](#):



The screenshot shows the 'Your Profile' page of the EchoNest developer portal. It has two tabs: 'Account Information' (selected) and 'Forum Subscriptions'. Under 'Account Information', there are three lines of masked text: 'Your API Key:', 'Your Consumer Key:', and 'Your Shared Secret:'. The 'Your API Key:' line is highlighted with a red rectangle. Below this are several input fields for personal information: Username, Email Address, First Name, Last Name, Company, Country, State/Province, Phone Number, Website URL, and Blog URL. The Email Address field is highlighted with a yellow background.

3. Copy and paste your API key to into `newcoder-api/src/newcoder-api/config.ini` like so:

[echonest]

api_key = YOUR_SUPER_SECRET_API_KEY

Spotify

Developer's site: [Spotify](#).

1. If you don't already, create a free account on Spotify.
2. Create a new application connected to your Spotify account [here](#) and click on 'Create an App':

Home

News

My Applications

Web API

HTML Widgets

iOS SDK (Beta)

Android SDK (Beta)

Spotify Apps


Libspotify SDK

Support

Design Resources


Terms of Use

My Applications




API Console - Development

Development for Interactive API console for developer.spotify.com




first

first testing app



API Console - Deployed

Deployed version of the web api console ***NOTE*** put OAuth client ID & secret within uWSGI's 'console.ini' file as an 'env'.



PyLadies Workshop

This is for the PyLadies workshop to teach how to use Spotify + EchoNest APIs with Python.

3. Fill in a name and a description for your application (doesn't matter, but something that makes sense to you).
4. After submitting, you'll be directed to your new app, where you'll see your `Client ID` and `Client Secret`.

PyLadies Workshop

Application Name *

PyLadies Workshop

Max 60 characters.

Application Description *

This is for the PyLadies workshop to teach how to use Spotify + EchoNest APIs with Python.

Describe your application in a few words, max 250 characters.

Website

Where the user may obtain more information about this application (e.g. <http://mysite.com>).

Client ID

Client Secret

Always store keys securely! **Regenerate** your client secret if you suspect it has been compromised!

Redirect URIs

ADD URI

White-listed addresses to redirect to after authentication success OR failure (e.g. <http://mysite.com/callback/>)

SAVE

CANCEL

DELETE

- Copy and paste the `Client ID` and `Client Secret` into your `config.ini` file under the `[spotify]` section, like so:

[echonest]

```
api_key = SUPER_SECRET_API_KEY
```

[spotify]

```
client_id = MY_SPOTIFY_CLIENT_ID
```

```
client_secret = MY_SPOTIFY_CLIENT_SECRET
```

GitHub

- [GitHub](#)

If you don't have an account at GitHub and you don't want to create one, write `None` next to `oauth =` in the `config.ini` file, so your `config.ini` looks like:

[echonest]

```
api_key = SUPER_SECRET_API_KEY
```

[spotify]

```
client_id = MY_SPOTIFY_CLIENT_ID
```

```
client_secret = MY_SPOTIFY_CLIENT_SECRET
```

[github]

```
oauth = None
```

If you **do** have an account (or want to create a free one, [here](#)):

1. Create a token for your application [here](#).
 - Write whatever you'd like in the Token description
 - **IMPORTANT:** be sure to have 'gists' checked!

- After you click `Generate Token`, copy the token and save it to `config.ini`, under the `[github]` section, like so:

[echonest]

```
api_key = SUPER_SECRET_API_KEY
consumer_secret = SUPER_SECRET_CONSUMER_KEY
shared_secret = SUPER_SECRET_SHARED_SECRET
```

[spotify]

```
client_id = MY_SPOTIFY_CLIENT_ID
client_secret = MY_SPOTIFY_CLIENT_SECRET
redirect_uri = http://localhost:5000/callback
```

[github]

```
oauth = GITHUB_PERSONAL_OAUTH_TOKEN
```

Before you leave for the evening!

Please grab a mentor, and show him/her the following:

1. Your `config.ini` file.
2. That you have a text editor installed.
3. With your terminal open:
 1. Show that you can get into your project directory, the `api-project`, via the `cd` command (ask for help if you're not sure!)
 2. Show that you can activate & deactivate your virtualenv
 3. **IMPORTANT STEP!** Within the activated virtualenv, run `pip list` for the mentor to review the output

4. Have some wine & cheese.
5. Feel awesome and prepared for Saturday's workshop!

See you in the morning!