Prepare your Dev Environment

Ing. Jimmy Figueroa

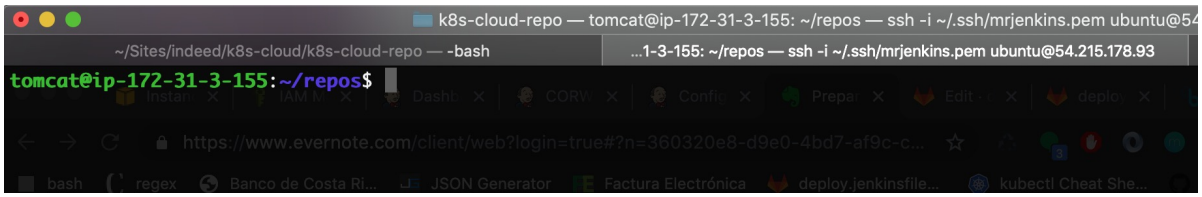
Introduction

The end result we're looking for is to have a «data development» environment ready to follow the rest of the classes of Big Data

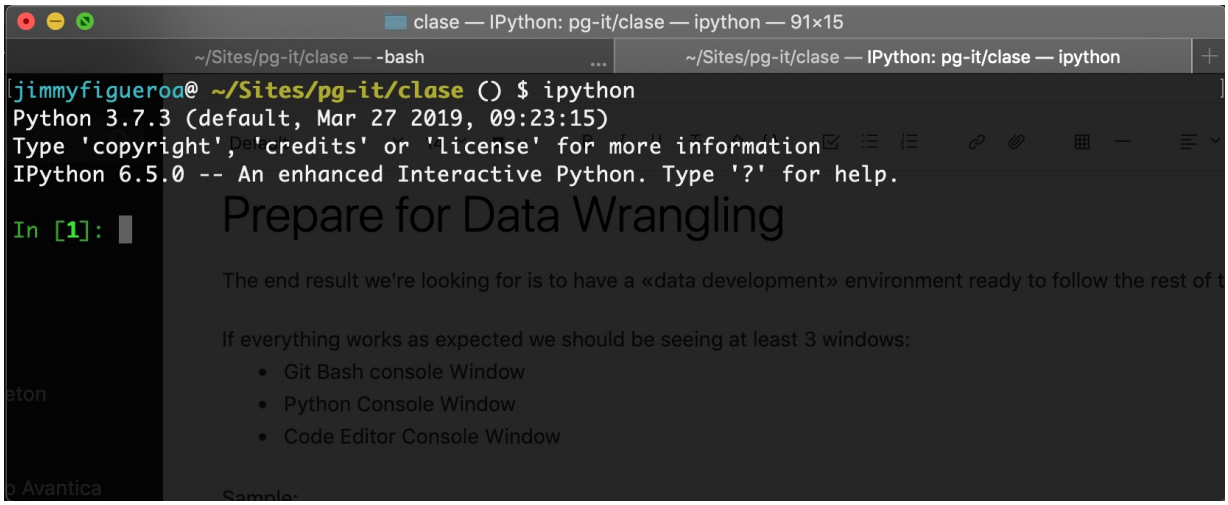
If everything works as expected we should be seeing at least 3 windows:

- Git Bash console Window
- Python Console Window
- Code Editor Console Window

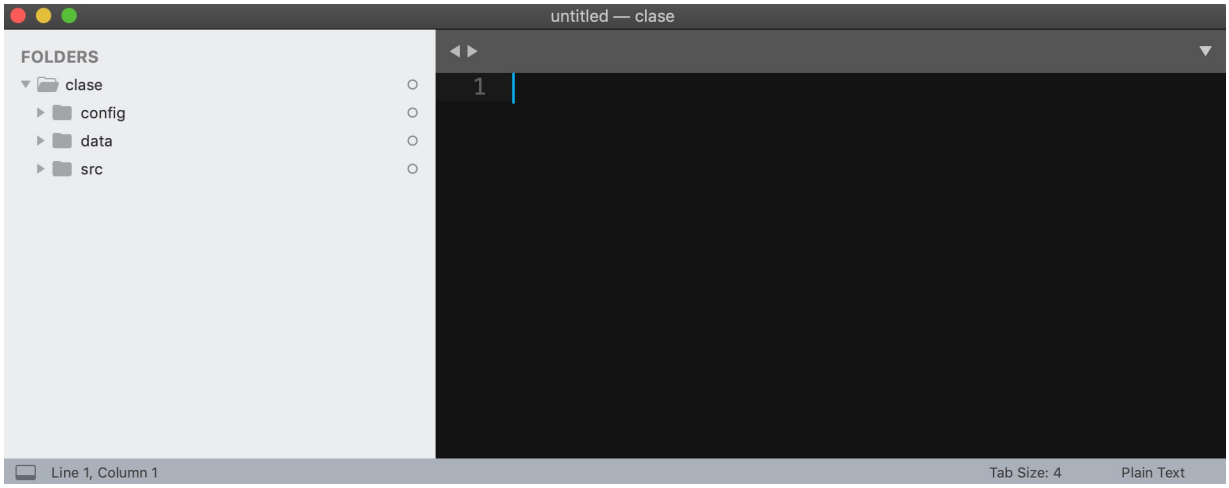
Sample git bash console:



Sample Python Console:



Sample Code Editor window:

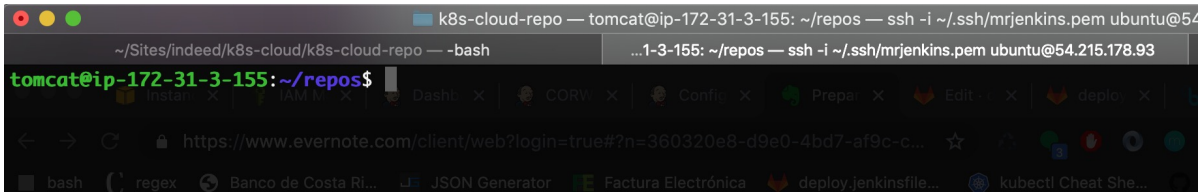


Install Git bash for Windows

- Install «Git Bash» for Windows from the following link

```
https://gitforwindows.org/
```

- Open «Git Bash» for Windows once. Make sure you Right click the icon and open it as Administrator. You should look at something like this:



Keep in mind the above is a «MacBook Pro» view of a Terminal Console. The Windows Console is similar but not identical.

At this point the most important fact is the your can see a `$` which is called the `prompt` of your `unix shell`.

From this point you can issue `unix` commands to a console in spite the fact you are running Windows Operating System.

The world of Data is by and large running in `unix` or `linux` boxes (a.ka. machines)

- What follows is a set of common unix command we use to get around building «data» services and solutions.

Commonly used Unix commands

---

prompt

Is the unix indicator by an `$` that you can type in any unix command.

```
$
```

Beware in some configurations the `prompt` may have been customized. Such is the case of Jimmy's machine, where the prompt looks like this:

```
jimmyfigueroa@ ~ () $
```

But it's the same thing, after the `$` you can type in any unix command.

pwd

Tells which is the current root directory

```
jimmyfigueroa@ ~ () $ pwd
/Users/jimmyfigueroa
jimmyfigueroa@ ~ () $
```

ls -ltr

List all files and folder in the current root directory. the ` -ltr` option lists the files/folders showing at the end the most recently updated files or folders.

```
jimmyfigueroa@ ~ () $ ls -ltr
total 104816
drwxr-xr-x    2 jimmyfigueroa  staff        64 Apr 15  2013 domi
-rw-r--r--    1 jimmyfigueroa  staff     42781 Jul 26  2013 twolines.png
-rw-r--r--    1 jimmyfigueroa  staff         0 Sep  2  2013 prueba-scd-2.architect~
-rw-r--r--    1 jimmyfigueroa  staff     11930 Sep  2  2013 prueba-scd-2.architect
-rw-r--r--    1 jimmyfigueroa  staff       814 Sep  2  2013 pruebascd2.sql
```

```
-rw-------    1 jimmyfigueroa  staff        43 Sep 22  2013 nohup.out
jimmyfigueroa@ ~ () $
```

mkdir <folder>

Creates a new folder (a.k.a. directory) from the current root directory.

grep <expression>

Filters output matching the expression.

|

It's called the «pipe» or «|». The pipe is used in unix to connect the output of one command with the next command.

In this example, after creating a directory or folder called «dummy», we list the files with `ls -ltr` and connect the output of that list

with a `grep` statement filters and shows only those files/folders with name matching `dummy`

```
jimmyfigueroa@ ~ () $ mkdir dummy
jimmyfigueroa@ ~ () $ ls -ltr | grep dummy
drwxr-xr-x    2 jimmyfigueroa  staff        64 Jun 10 14:31 dummy
jimmyfigueroa@ ~ () $
```

cd

Changes the root to a different directory. In the example we changed root directory multiple times to arrive at a root called `/Users/jimmyfigueroa/Sites/pg-it/clase`

```
jimmyfigueroa@ ~ () $ cd Sites
jimmyfigueroa@ ~/Sites (new_ux) $ cd pg-it
jimmyfigueroa@ ~/Sites/pg-it () $ cd clase
jimmyfigueroa@ ~/Sites/pg-it/clase () $ pwd
/Users/jimmyfigueroa/Sites/pg-it/clase
jimmyfigueroa@ ~/Sites/pg-it/clase () $
```

In order for the students to «build» data solutions, we have suggested the following initial data structure from a given root directory:

```
$ pwd
/Users/jimmyfigueroa/Sites/pg-it/clase
$ mkdir config
$ mkdir data
$ mkdir src
```

touch

Creates a new empty file with a name. Like this:

```
$ pwd
/Users/jimmyfigueroa/Sites/pg-it/clase
$ touch config/config.yml
$ touch data/jugadores.csv
$ touch src/sele.py
```

tree

Some unix have this command «tree» that list files/folders in a tree fashion like this:

```
jimmyfigueroa@ ~/Sites/pg-it/clase () $ tree
.
├── config
|   └── config.yml
├── data
|   └── jugadores.csv
└── src
    └── sele.py
3 directories, 3 files
jimmyfigueroa@ ~/Sites/pg-it/clase () $
```

Code Editor

We have suggested to use «Sublime Text 3», or «Atom» or «Vscode».
All of those are open source, pick one, google it, install it and open it.

Python Console

From the `$` prompt you can access the `python console` like this:

```
$ ipython

Python 3.7.3 (default, Mar 27 2019, 09:23:15)
Type 'copyright', 'credits' or 'license' for more information
IPython 6.5.0 -- An enhanced Interactive Python. Type '?' for help.
In [1]:
```

If for any reason you can't access `ipython`, it could be your Python package is installed, but ipython - the wrapper/launcher -- is missing. To check whether this is the case try running it like this:

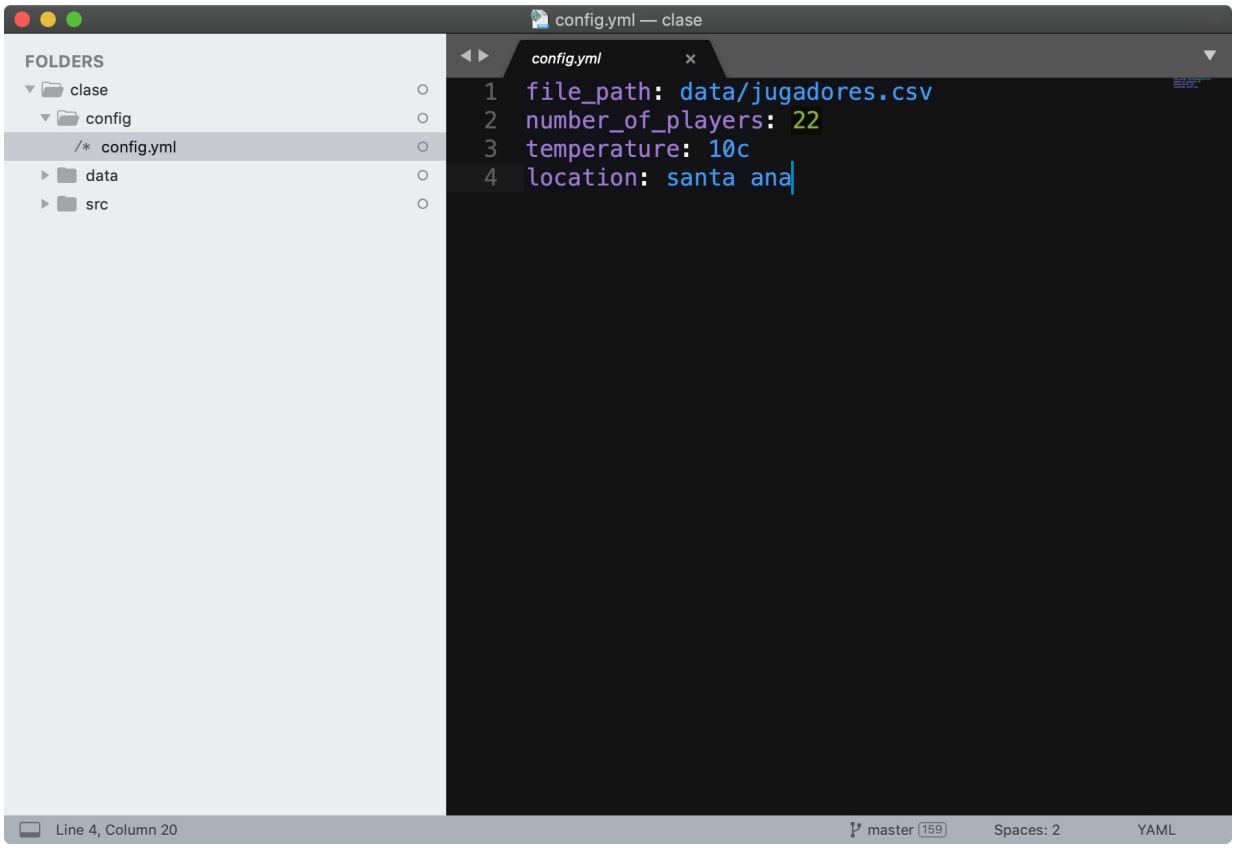
```
$ python -m IPython
Python 2.7.9 (default, Feb 10 2015, 03:28:08)
Type "copyright", "credits" or "license" for more information.


IPython 3.7.3 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.


In [1]:
```

If the above fails try accessing `iPython` from your `Anaconda` dashboard in your Windows System.
If none of this works, try installing `ipython` in your windows with instructions in this link `https://ipython.org/install.html`

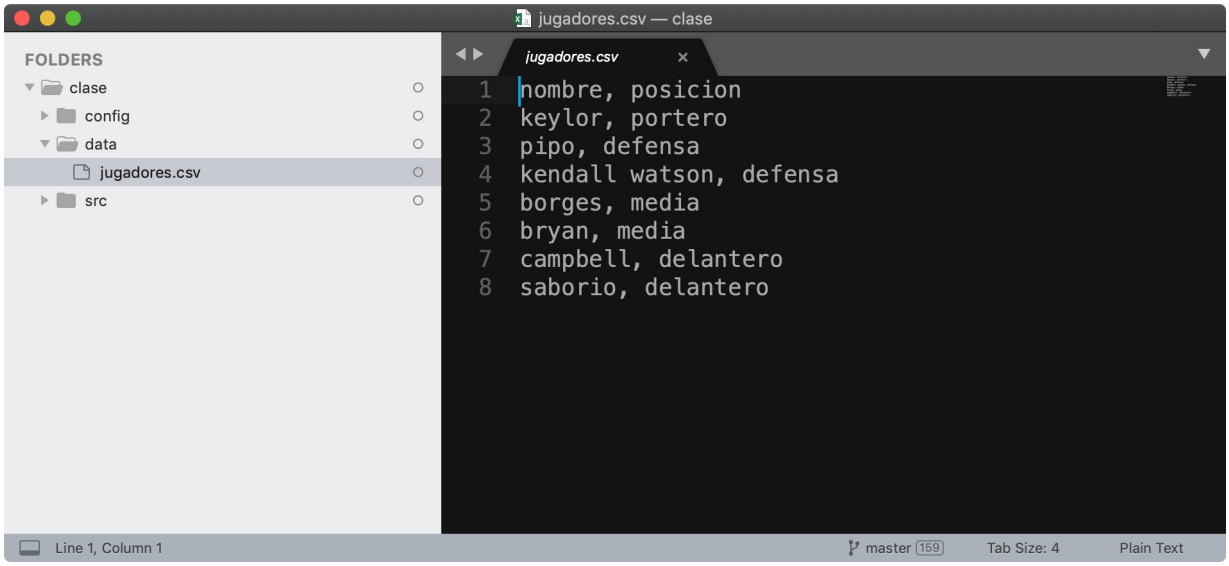
Initial Code

For the purpose of preparation for the upcoming class, we wrote some initial code.

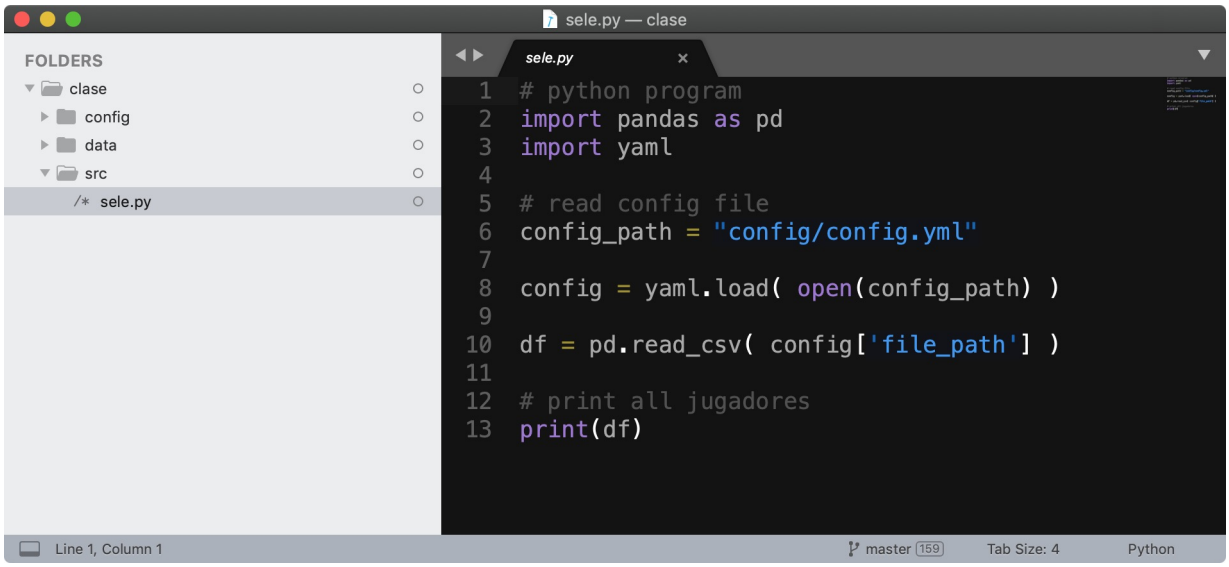
`./config/config.yml` is a file that contains only configuration settings. Each entry in this YAML file consist of one `key` and one `value` separated by one `:` colon as in the image bellow.



`/data/jugadores.csv` contains a list of comma-separated values for jugadores de futbol, as in the image bellow:



Lastly, `./src/sele.py` is the python program we wrote to put all together and have a print out in screen of all the `records` for `jugadores.csv` like this:



Testing your Code

One way to test your code is to just «copy/paste» your code in «sele.py» from your code editor into you «python console» like this:

```
clase — IPython: pg-it/clase — ipython — 80×32
~/Sites/pg-it/clase — -bash    ~/Sites/pg-it/clase — IPython: pg-it/clase — ipython

In [1]: # python program
   ...: import pandas as pd
   ...: import yaml
   ...:
   ...: # read config file
   ...: config_path = "config/config.yml"
   ...:
   ...: config = yaml.load( open(config_path) )
   ...:
   ...: df = pd.read_csv( config['file_path'] )
   ...:
   ...: # print all jugadores
   ...: print(df)
   ...:
   ...:
          nombre    posicion
0          keylor     portero
1            pipo     defensa
2  kendall watson     defensa
3          borges       media
4           bryan       media
5        campbell   delantero
6         saborio   delantero

In [2]:
```

Another better way of testing your code is to type the following command from the «root» directory of your «git bash shell» like this:

```
clase — IPython: pg-it/clase — -bash — 105×21
~/Sites/pg-it/clase — IPython: pg-it/clase — -bash    ~/Sites/pg-it/clase — IPython: pg-it/clase — ipython

jimmyfigueroa@ ~/Sites/pg-it/clase () $ ipython src/sele.py
          nombre    posicion
0         keylor     portero
1           pipo     defensa
2  kendall watson    defensa
3         borges       media
4          bryan       media
5       campbell   delantero
6        saborio   delantero
jimmyfigueroa@ ~/Sites/pg-it/clase () $
```

If for some reason you could only access your IPython from the «anaconda» dashboard, you may try this to run your code:

```
clase — IPython: pg-it/clase — ipython — 94×22
~/Sites/pg-it/clase — IPython: pg-it/clase — -bash    ~/Sites/pg-it/clase — IPython: pg-it/clase — ipython

In [2]: ! ipython src/sele.py
          nombre    posicion
0         keylor     portero
1           pipo     defensa
2  kendall watson    defensa
3         borges       media
4          bryan       media
5       campbell   delantero
6        saborio   delantero

In [3]:
```

Extended Reading

You may learn more of common or basic unix command from the following links:

https://www.tipsandtricks-hq.com/basic-unix-commands-list-366
https://www.softwaretestinghelp.com/unix-commands/
https://www.techonthenet.com/unix/basic/ls.php