# Python Boot-camp



## Overview

### Workshop Video

`\\bjg.cn.ao.ericsson.se\Usertmp$\pyCamp\Video`

### Workshop Keynotes and Codes

`\\drive.swdp.me\learning\pyCamp\codesAndKeynotes`

| Time | Task | Outcome | Reference |
| --- | --- | --- | --- |
| W0 | Prepare environment, join the slack channel for discussion or learn from historical discussion. | A ready to use python envrionment. | This document |
| W1 | Task 0, build a cli tool | Learn basic concept, build a command line tool, run it. | |
| | Task 1, build a web application. | Reach more python concepts, build a website, build a webapp with same function of task | |
| | Task 2, build a blockchain. | Start building your first OOP python project, organize a bigger flask project. | |
| W2 | Task 3, build a microservice project | TBD | |
| | Task 4, build an openstack project | TBD | |

# Preparation

Slack channel: #py-camp

## Environment setup

a Linux VM (Ubuntu) will do, reference: https://askubuntu.com/questions/142549/how-to-install-ubuntu-on-virtualbox

- 3G ram
- 1-2 CPU

## A just working editor

`Sublime text` / `Atom` / `VS code` will do the job, I was using `Sublime Text 3`

reference: http://tipsonubuntu.com/2017/05/30/install-sublime-text-3-ubuntu-16-04-official-way/

## How to run your p ython code?

**Preparation**:

to avoid anything stops you from the beginning let's just use python3 directly, if you know how to use `virtualenv`, use it instead, if you don't know `virtualenv` do it as below.

- install pip for python3, upgrade it then:

```
$ sudo apt-get install python3-pip  -y
$ sudo pip3 install  --upgrade pip
```

- install `ipython` as your python playground console

```
$ sudo pip3 install ipython
```

**Hello world!** from ipython

You could use it in one terminal for debugging for always, it's better than the python console in some ways.

```
$ ipython3
Python 3.5.2 (default, Nov  23 2017, 16:37:01)
Type 'copyright', 'credits' or 'license' for more information
IPython 6.2.1 -- An enhanced Interactive Python. Type  '?' for help.

In [1]: print ("hello world!")
hello world!
In [2]: exit
$
```

**Hello world!** from your first python script

```
$ cat hello_world.py
print ("Hello world!")

$ python3 hello_world.py
Hello world!
```

Congratulations! One more step left:

## Go through the tutor

reference: https://docs.python.org/3/tutorial/

or: http://www.pythondoc.com/pythontutorial3/

# Tasks

## Task 0, create a cli tool

Let's create tool to parse a `nova/nova.conf` file in order to get expected values, like `my_ip` , if what `virt_type` etc..

What needs to be done?

- A function to handle the actual input.
- make it handle arguments from command line
- Let's make it professional
- Use `docopt` to simplify it! reference: http://docopt.org

Further study:

- Shebang ref: https://en.wikipedia.org/wiki/Shebang_%28Unix%29
- `virtualenv`
- parse from or write to json, excel, xml, csv etc ?

    - built-in library: https://docs.python.org/3/library/index.html
    - Awesome Python: https://github.com/vinta/awesome-python
- Doing things for servers? paramiko / ansible
- exceptions: http://www.runoob.com/python/python-exceptions.html

**After class task:**

- create a cli tool

## Task 1, create a website

This task is to create a web app with flask, following the flask-mega-tutorial first three chapters.

reference:

- innoSearch project: https://github.com/littlewey/yet-another-GSC-C-lighthouse
- ldif-compare: https://github.com/littlewey/ldif-compare

**After class task:**

- create a web tool, which could reuse the main function of Task 0.

## Task 2, create a `blockchain` (ICO yourself)

> Our task demo is highly reusing code and even some words in https://hackernoon.com/learn-blockchains-by-building-one-117428612f46 (many thanks to Daniel van Flymen), some enhancements were added on top of that here including sender address verification, and some script clients.

In this task, we will create a simple part of blockchain application like `bitcoin`.

Things we will go though:

- Some object-oriented programing (not just call class from a python module)
- http request
- flask as http api endpoints

**After class task:**

- create the blockchain app following the tutorial, study the code (as explained in class), if possible improve it.

## Task 3, create a big project(micro service enabled)

> haproxy, etcd, k8s etc...

To be continued in next presenting week.

## Task 4, create a tool based on oslo

> Openstack related

To be continued in next presenting week.