



Vidyut '20

Python for Cloud and Embedded Systems

Lab Manual



Vysakh Pillai & Abhijeet Prem



Python for Cloud and Embedded Systems @ Vidyut '20

Lab Manual

Introduction

This lab manual is developed to get you going with the hands-on part of “Python for cloud and embedded systems”. It is split into labs and exercises within each lab session. The contents are developed to accelerate the engagement of the participants in the session. Consequently, information in this manual is not elaborate. Feel free to talk to one of the lab helpers or presenters in case you are stuck.

Contents

Python for Cloud and Embedded Systems @ Vidyut '20.....	1
Lab Manual	1
Introduction	1
Lab 1 – Getting Started	3
Exercise 1.1 – Getting started with Git.	4
Exercise 1.2 – Getting started with Jupyter notebooks.....	5
Exercise 1.3 – Getting started with VS Code environment.....	6
Lab2 – Understanding networking basics with Python.	7
Exercise 2.1 – Fetching a raw webpage	7
Exercise 2.2 – Fetching a raw webpage – with code	7
Exercise 2.3 – Creating your own webpage (not a website).....	7
Exercise 2.4 – Embedding a web page withing Jupyter notebook	7
Exercise 2.5 – Fetching raw webpage – production mode.....	7
Lab 3 – Setting up an environment.....	8
Exercise 3.1 – <code>virtualenv</code> Setup	8
LAB4 – Setting up your web server.....	9
Exercise 4.1 – Flask	9
Observe:.....	9
DIY	9
LAB 5 – A simple web API	10
Exercise 5.1 – Web API with URL parameters	10
DIY	10
Exercise 5.2 – Google QR Code Web API	10
LAB 6 – Python for embedded systems	11
Exercise 6.1 – Hello Blinky World	11
Exercise 6.2 – Interact with the board.....	12
Appendix 1	13
Tools to be installed for Lab sessions	13

Lab 1 – Getting Started

In this lab, we will get the workspace setup, get an understanding of the tools we will be using in this class and ensure that our tools are all working fine.

At the end of this lab, you would have:

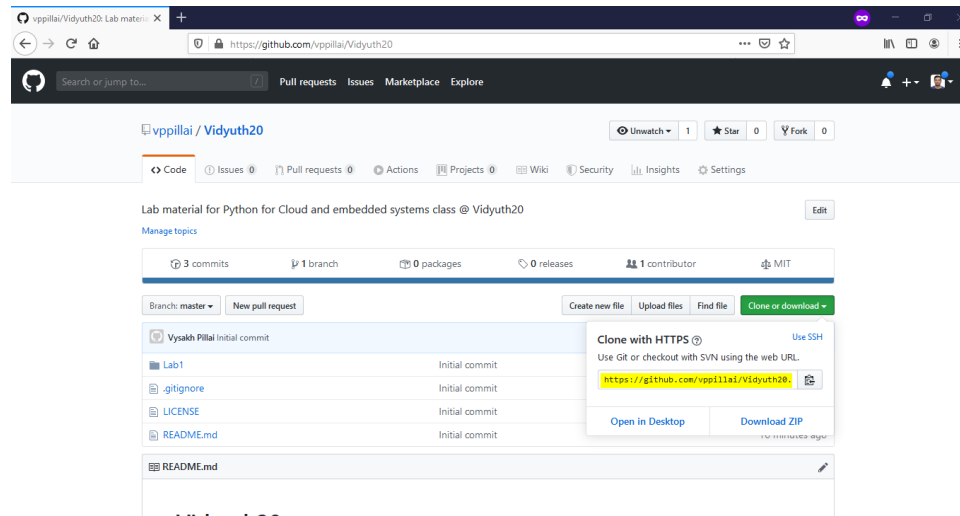
- 1) Setup a python development environment using Jupiter notebooks as well as Visual studio Code
- 2) Downloaded all the collateral required for the complete session from Github
- 3) Executed some basic examples in your development environment.

Exercise 1.1 – Getting started with Git.

The theory session associated with this exercise will cover Git Basics.

All the source files for this class is stored in Github. To get them:

- 1) Navigate to <https://github.com/vppillai/Vidyut20>
- 2) Copy the clone link

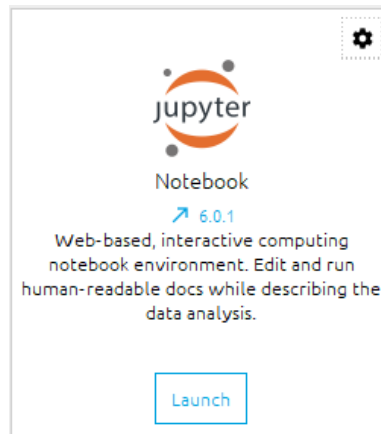


- 3) Create a local folder within the “Documents” folder in your PC and navigate to it.
 - a. Make a note of this folder path since we will be using it throughout this class.
- 4) From a command prompt, issue the following command
 - a. `git clone https://github.com/vppillai/Vidyut20.git`
- 5) Now you have all the necessary files for this session in your working folder.

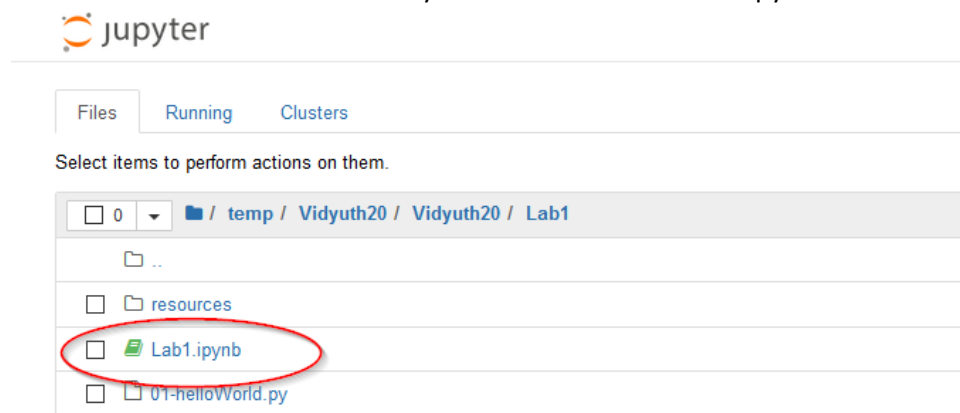
Exercise 1.2 – Getting started with Jupyter notebooks.

A Jupyter notebook based environment is very close to your R&D setup.

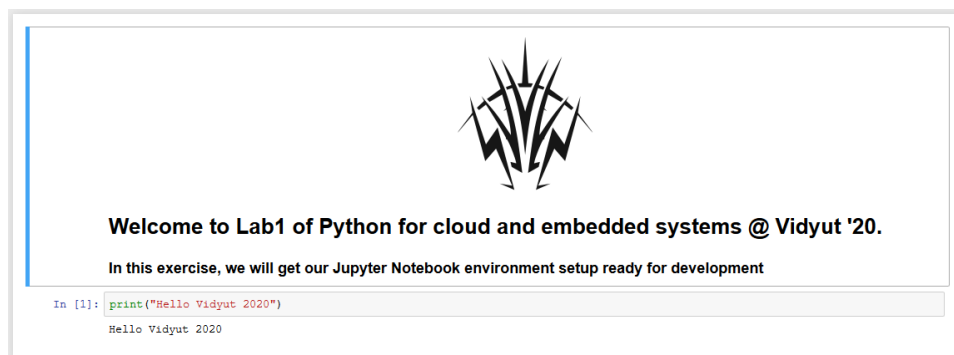
- 1) Open anaconda navigator
- 2) Launch Jupyter Notebook




- 3) Navigate to Lab1 folder within the source files you cloned from *Git* from Jupyter window



- 4) Click on Lab1.ipynb and observe the output



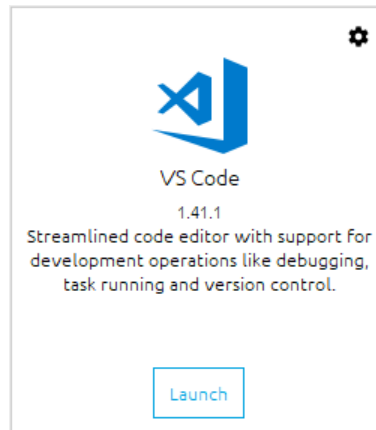
- 5) Try modifying the code and text and then click  **Run** to see your changes in action.

Now, you have setup your Jupyter notebook environment for development.

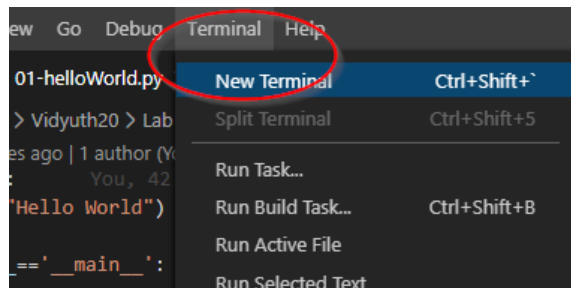
Exercise 1.3 – Getting started with VS Code environment.

In this exercise, we will setup the VS Code environment for development. This is closer to a development setup for production

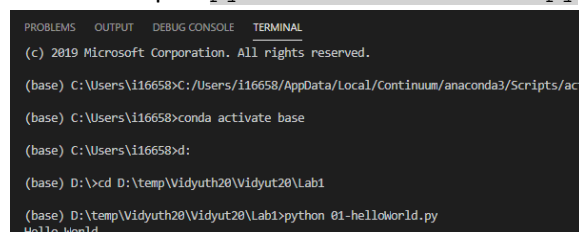
- 1) Open Anaconda Navigator and click VS Code.



- 2) Open `Vidyut20\Lab1\01-helloWorld.py`
- 3) Open a new terminal



- 4) Navigate to your working folder in the terminal
- 5) Issue the command and observe the output: `python 01-helloWorld.py`



Now, you have setup your VS Code environment for development

Lab2 – Understanding networking basics with Python.

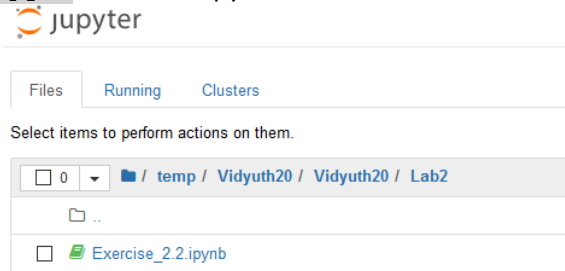
In this lab, we will experiment with some of the basic networking concepts that we discuss in the theory class. To start with, we will understand how a browser works while loading a webpage for you.

Exercise 2.1 – Fetching a raw webpage

- 1) Navigate to google.com
- 2) Right click on the page and select “view page source”

Exercise 2.2 – Fetching a raw webpage – with code

- 3) Open the `Exercise_2.2.ipynb` file from Jupyter Notebook



- 4) Execute it and observe the result.

Observations will be discussed in the session

Exercise 2.3 – Creating your own webpage (not a website 😊)

- 1) Open `Vidyut20\Lab2\resources\index.html` in a browser and observe the output.
- 2) Open it in a text editor, modify it and observe the output in the browser.

Exercise 2.4 – Embedding a web page withing Jupyter notebook

- 1) Open `Exercise_2.4.ipynb` and observe the output.

Exercise 2.5 – Fetching raw webpage – production mode.

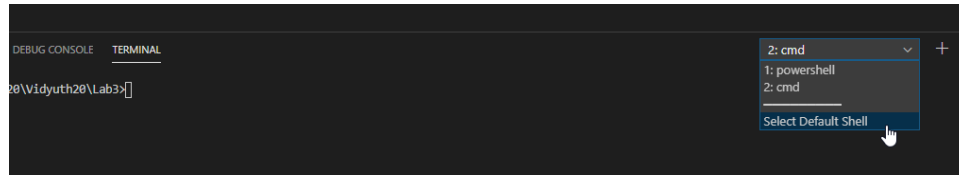
- 1) Open `02-getPage.py` in VS Code.
- 2) Execute the file in a terminal.
- 3) Observe the difference in code flow and execution.

Lab 3 – Setting up an environment

You do not want your development or production setup to be affected by an update to the system. We use `virtualenv` to create a sandboxed environment for your python script to execute.

Exercise 3.1 – `virtualenv` Setup

- 1) Change the default terminal in VS Code to `cmd`



- 2) Install `virtualenv` by issuing the command
`pip install virtualenv`
- 3) Navigate to Lab3 Folder in your working directory.
- 4) Setup an environment called “ENV” by issuing the command :

```
virtualenv ENV
```

- 5) Activate the environment by issuing the command and note the change in environment.

```
ENV\Scripts\activate
```

```
(base) D:\temp\Vidyuth20\Vidyuth20\Lab3>ENV\Scripts\activate  
(ENV) (base) D:\temp\Vidyuth20\Vidyuth20\Lab3>
```

- 6) Explore the ENV folder and check out the contents.
 - a. `Lab3\ENV\Lib\site-packages` is especially interesting.

We will be using this concept heavily in subsequent lab sessions

LAB4 – Setting up your web server

In this exercise, we will setup a development web server and launch a simple “website” from it. Eventually, we will build a simple API with it.

Exercise 4.1 – Flask

- 1) Navigate to `Vidyut20\Lab4` and open `lab4-BasicServer.py` in VS Code.
- 2) Open a terminal and setup the environment using the command:
`lab4_env\Scripts\activate`
- 3) Run the flask app by using:
`flask run`

```
(lab4_env) (base) D:\temp\Vidyuth20\Vidyuth20\Lab4>flask run
* Serving Flask app "lab4-BasicServer.py"
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

- 4) Navigate to the address displayed in the console in a browser.

Observe:

- Look in the following files
`Vidyut20\Lab4\app__init__.py`
`Vidyut20\Lab4\app\routes.py`
- Look at the source file of the page served by flask and notice that it is plain text.

DIY

- Try modifying the contents of the page you are served from Flask.

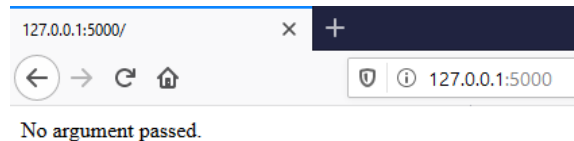
The overall flask flow will be covered in the theory session

LAB 5 – A simple web API

In this exercise, you will be introduced to the most simple form of web API calls – URL parameters.

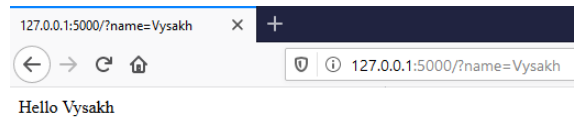
Exercise 5.1 – Web API with URL parameters

- 1) Navigate to `Vidyut20\Lab5` and open `lab5-ArgServer.py` in VS Code.
- 2) Open a terminal and setup the environment using the command:
`lab5_env\Scripts\activate`
- 3) Run the flask app by using:
`flask run`
- 4) Navigate to the address displayed in the console in a browser and observe the output.



- 5) Modify the Browser URL to:

`127.0.0.1:5000?name=Vysakh`



DIY

- Try adding another parameter to the server code

Exercise 5.2 – Google QR Code Web API

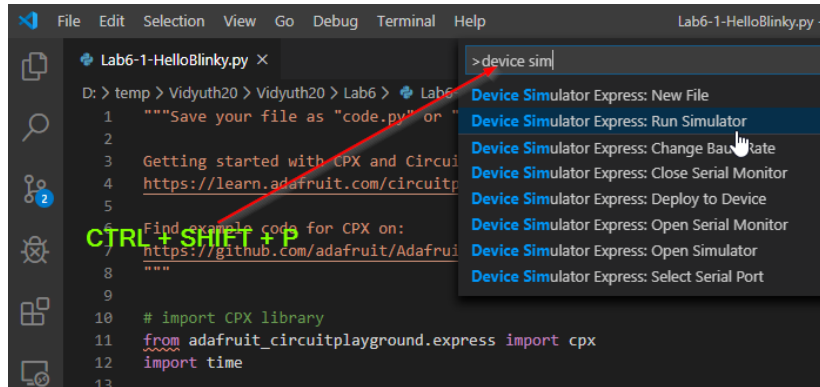
- 1) Execute `Vidyut20\Lab5\Lab5.ipynb` and follow the steps in the notebook.

LAB 6 – Python for embedded systems

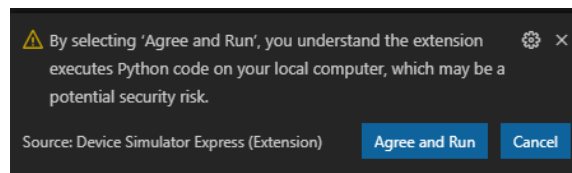
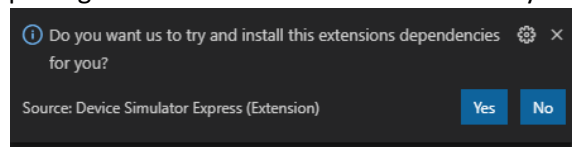
In this exercise we will run python on an embedded device using a simulator.

Exercise 6.1 – Hello Blinky World

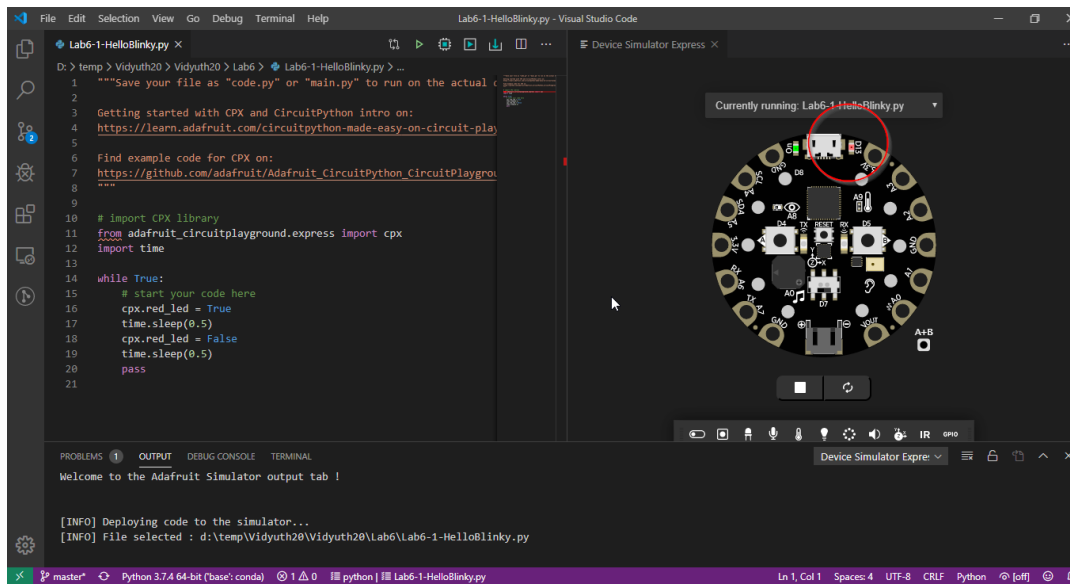
- 1) Launch VS Code outside anaconda.
- 2) Open `Vidyut20\Lab6\Lab6-1-HelloBlinky.py`
- 3) CTRL+SHIFT+P and enter device simulator. Select Run Simulator



- 4) Select “No” if prompted for package installation. This will be followed by a service agreement.



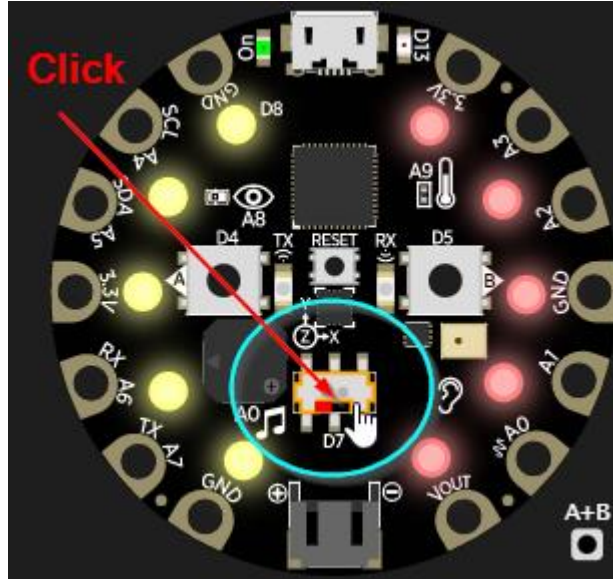
- 5) You will see the code executing in the simulator and blinking the onboard LED



- 6) Open the following files and explore python in action in the simulator.
 - o `Lab6-1a-NeoPixels.py`

Exercise 6.2 – Interact with the board

- 1) Load `Lab6-3-Switch.py` into the simulator and execute as before.
- 2) The animation will change with a button press.



Appendix 1

Tools to be installed for Lab sessions

- 1) *Git* for Windows: <https://github.com/git-for-windows/git/releases/download/v2.25.0.windows.1/Git-2.25.0-64-bit.exe>
- 2) VS Code stable: <https://aka.ms/win32-x64-user-stable>
- 3) Following extensions should be installed in VS Code (outside anaconda).
 - a. ms-python.python
 - b. ms-python.anaconda-extension-pack
 - c. ms-python.devicesimulatorexpress
 - i. make sure Exercise 6-1 can be executed. This will install all required dependencies.
 1. Launch VS Code outside anaconda
 2. Open a new terminal
 3. `pip install pywin32`
 4. `pip install python-socketio`
 5. `pip install requests`
 6. `pip install applicationinsights`
 7. `pip install playsound`
- 4) Python 3.8 : <https://www.python.org/ftp/python/3.8.1/python-3.8.1.exe>
- 5) Anaconda 2019.10 (Python 3.7): https://repo.anaconda.com/archive/Anaconda3-2019.10-Windows-x86_64.exe
- 6) Head over to this link to install the Device Simulator Express:
<https://marketplace.visualstudio.com/items?itemName=ms-python.devicesimulatorexpress>