PYNQ Getting Start

(Based On KD240)

AMD

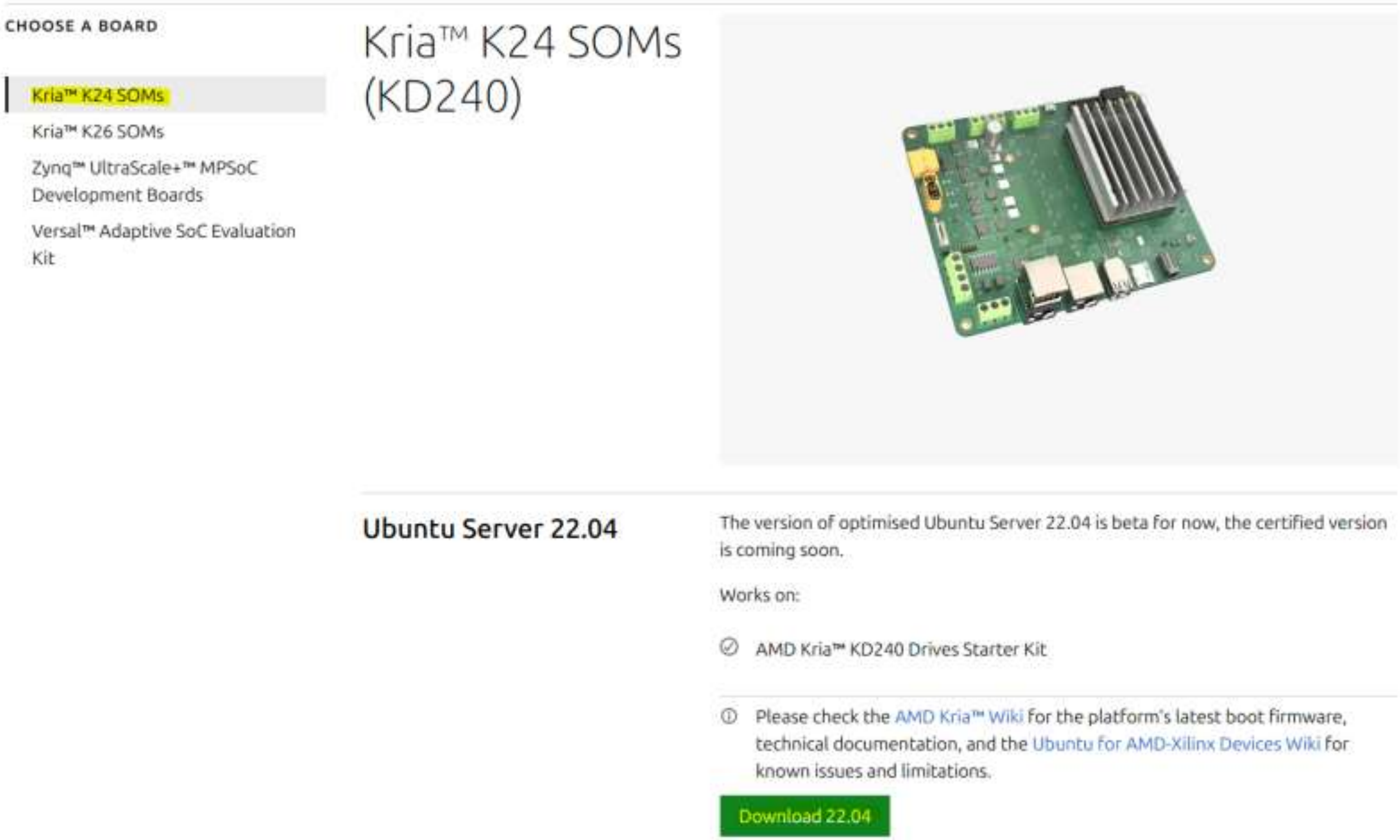
together we advance_

Install Ubuntu on AMD

PYNQ currently only supports ZYNQ Based and Versal series(include KRIA , Alveo).

PYNQ is a framework designed for the Ubuntu environment, so we must first install Ubuntu on the development board.

Install Ubuntu on AMD | Ubuntu



CHOOSE A BOARD

- Kria™ K24 SOMs
- Kria™ K26 SOMs
- Zynq™ UltraScale+™ MPSoC Development Boards
- Versal™ Adaptive SoC Evaluation Kit

Kria™ K24 SOMs (KD240)

Ubuntu Server 22.04

The version of optimised Ubuntu Server 22.04 is beta for now, the certified version is coming soon.

Works on:

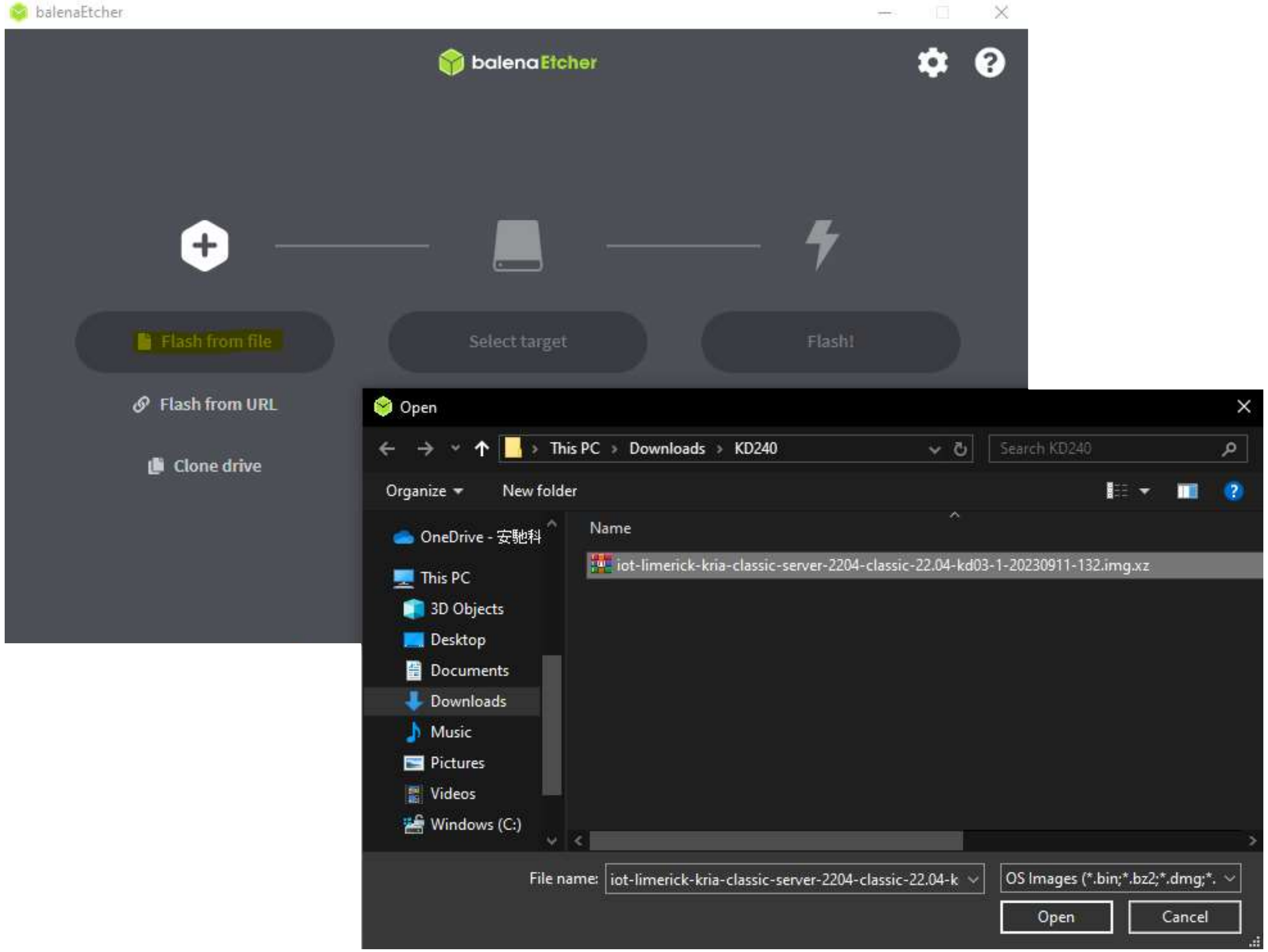
✓ AMD Kria™ KD240 Drives Starter Kit

ⓘ Please check the AMD Kria™ Wiki for the platform's latest boot firmware, technical documentation, and the Ubuntu for AMD-Xilinx Devices Wiki for known issues and limitations.

Setting Up the SD Card Image (Ubuntu)

Follow the instructions in the tool and select the downloaded image to flash onto your microSD card.

Setting up the SD Card Image (xilinx.com)



iot-limerick-kria-classic-server-2204-classic-22.04-kd03-1-20230911-132.img.xz

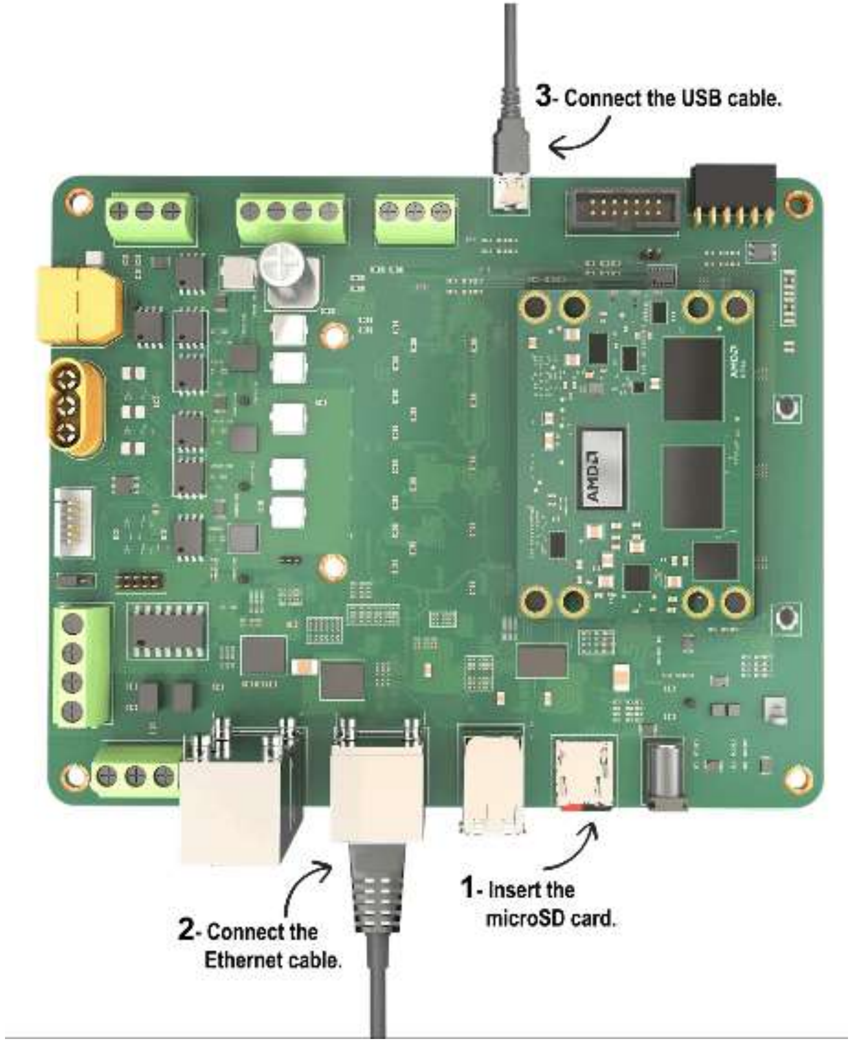
File name: iot-limerick-kria-classic-server-2204-classic-22.04-k

OS Images (*.bin;*.bz2;*.dmg;*.

AMD
together we advance_

Board Setup

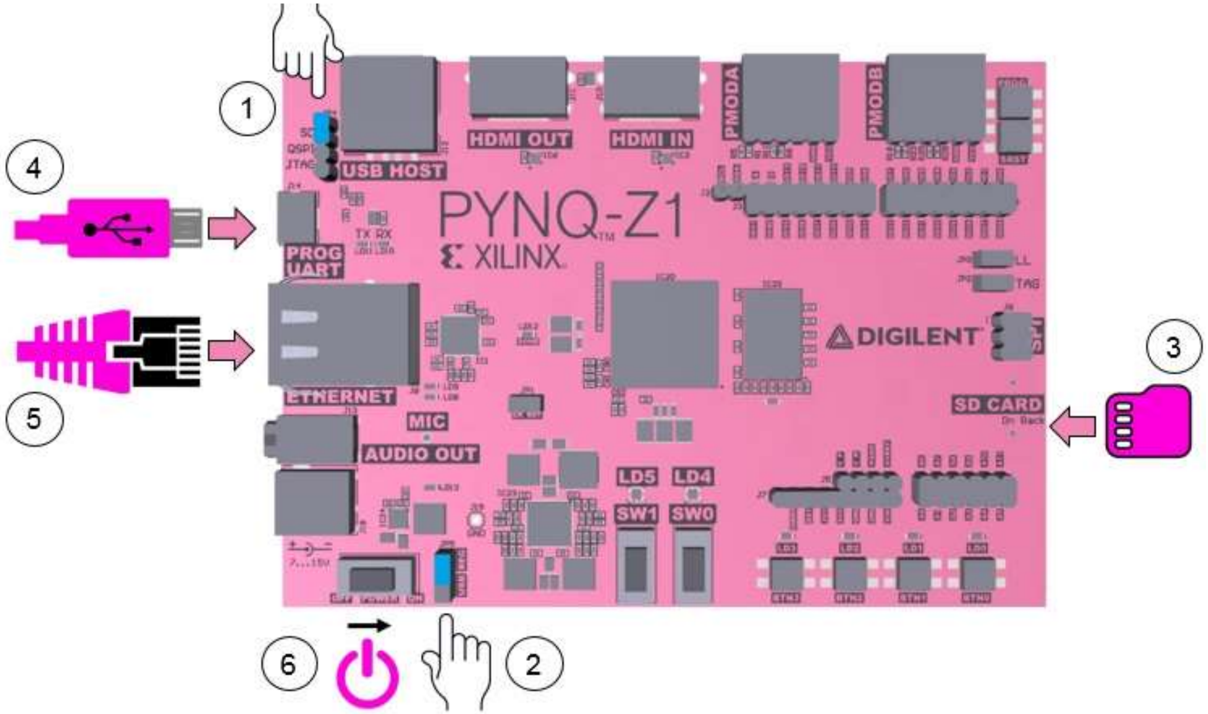
- Insert the SD card with the Ubuntu image, and ensure that it is connected to the external network.
- UART is optional, but it is helpful for accelerating development and debugging.



3- Connect the USB cable.

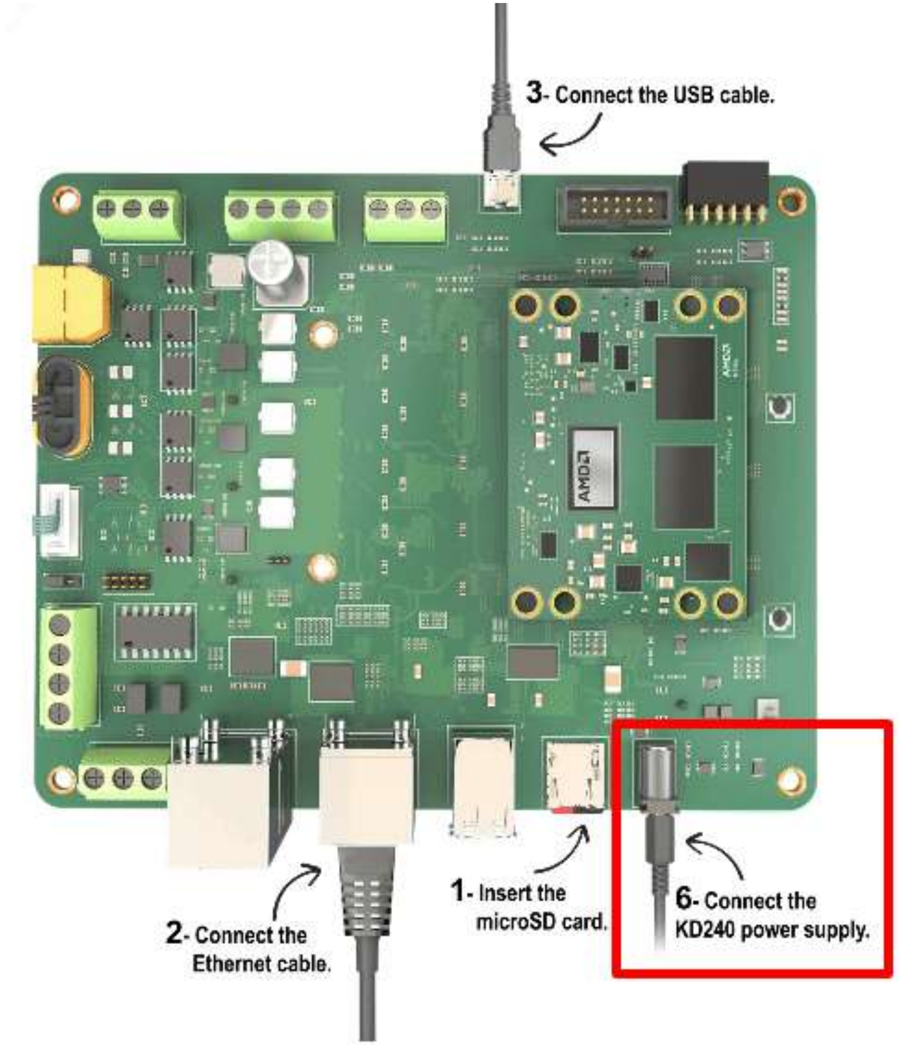
1- Insert the microSD card.

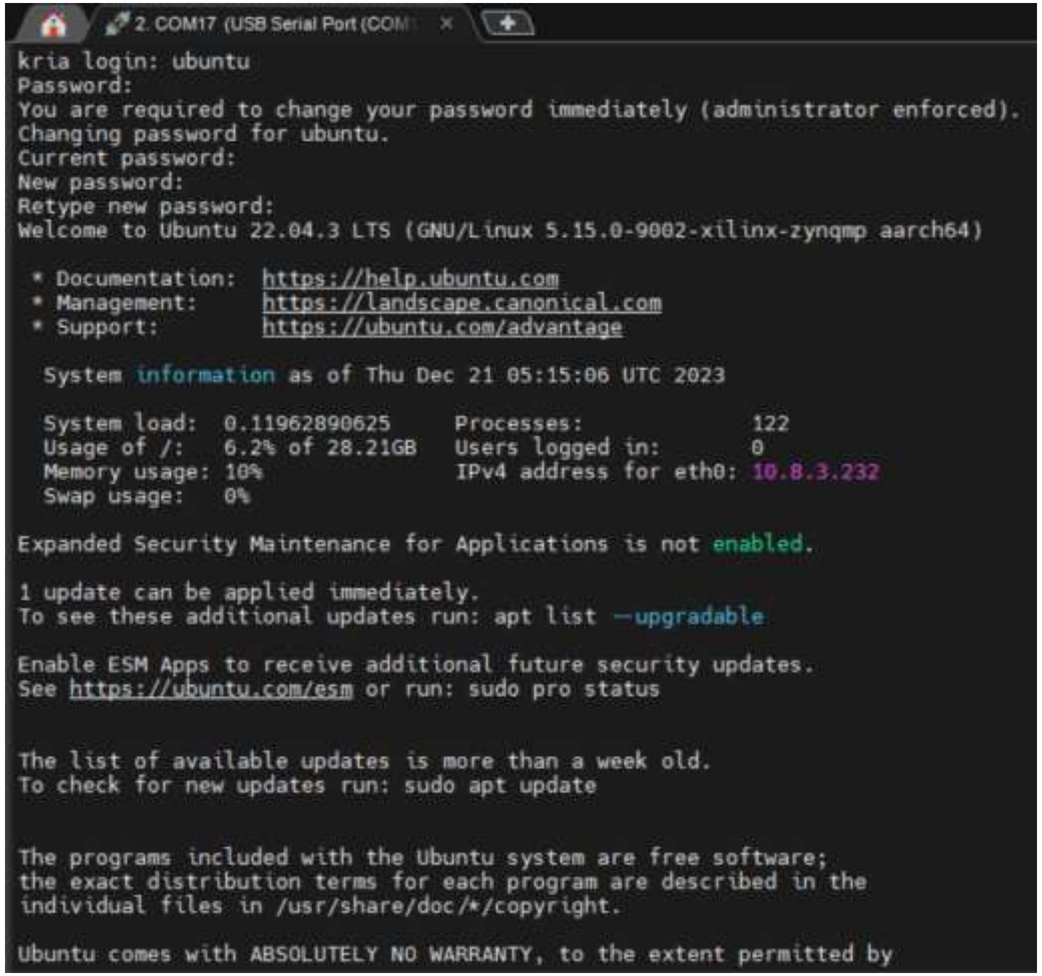
2- Connect the Ethernet cable.



Board Setup

Power on and communicate with the development board through UART (choose a preferred serial port debugging app.


3- Connect the USB cable.
2- Connect the Ethernet cable.
1- Insert the microSD card.
6- Connect the KD240 power supply.



```
kria login: ubuntu
Password:
You are required to change your password immediately (administrator enforced).
Changing password for ubuntu.
Current password:
New password:
Retype new password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-9002-xilinx-zynqmp aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Thu Dec 21 05:15:06 UTC 2023

  System load:  0.11962890625     Processes:             122
  Usage of /:   6.2% of 28.21GB   Users logged in:       0
  Memory usage: 10%               IPv4 address for eth0: 10.8.3.232
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
```

Install PYNQ

The login username and password are both "ubuntu" After the first login, you will be prompted to change your password.

After logging in, be sure to execute the following command:

```
sudo apt-get update

sudo apt-get upgrade
```

Next, refer to the following URL.

GitHub - Xilinx/Kria-PYNQ: PYNQ support and examples for Kria SOMs

Execute the following command:

```
git clone https://github.com/Xilinx/Kria-PYNQ.git

cd Kria-PYNQ/

sudo bash install.sh -b KD240
```

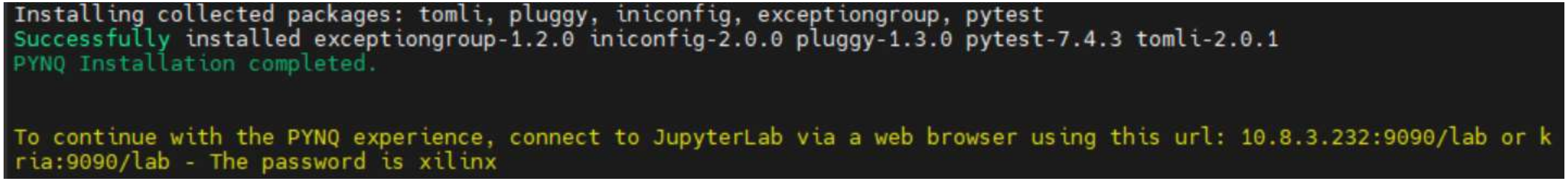
This process will take approximately one hour.

For non-Kria series development boards, you need to download the corresponding image file or build yourself.

GitHub - Xilinx/sdbuild : PYNQ Image Building

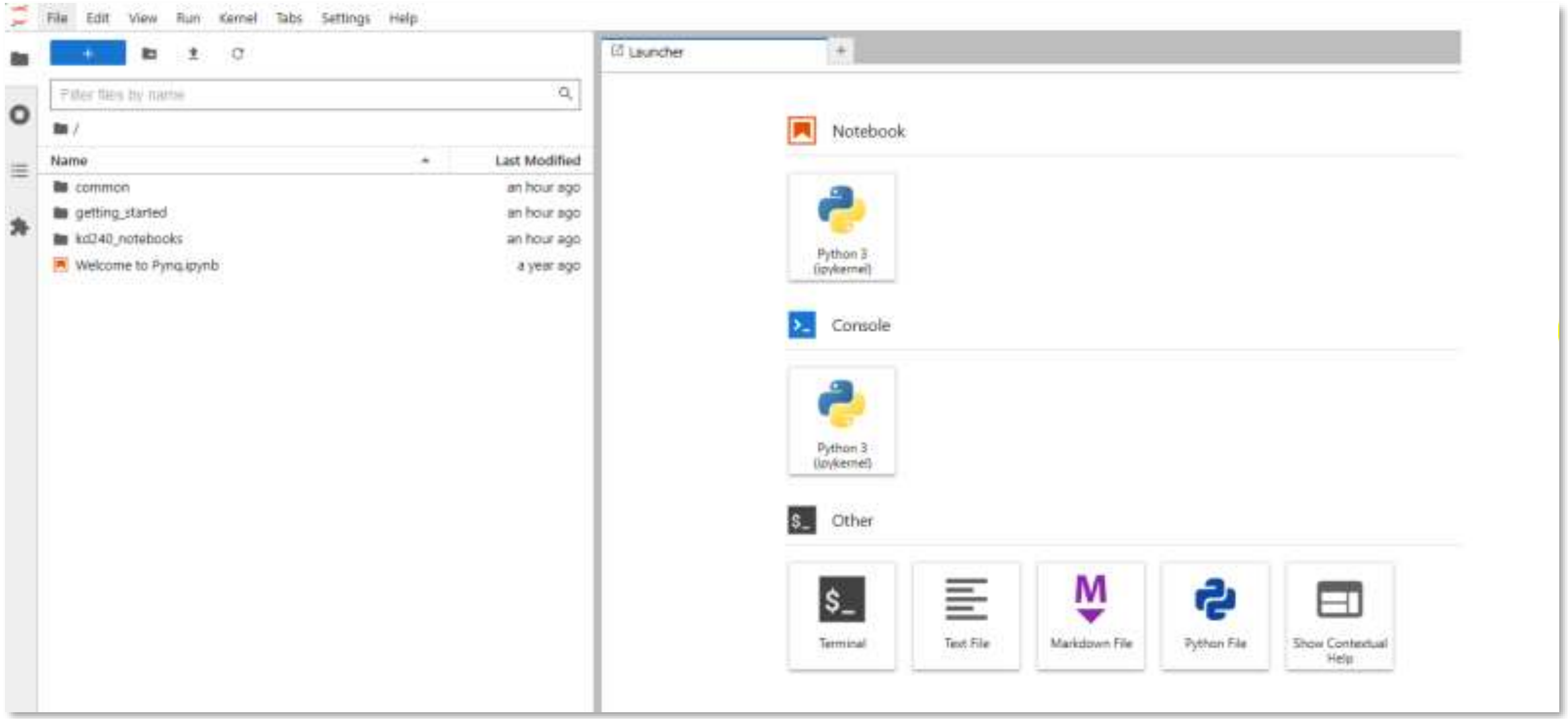
Connecting to Jupyter Notebook

After a successful installation, you will see the following message.



```
Installing collected packages: tomli, pluggy, iniconfig, exceptiongroup, pytest
Successfully installed exceptiongroup-1.2.0 iniconfig-2.0.0 pluggy-1.3.0 pytest-7.4.3 tomli-2.0.1
PYNQ Installation completed.


To continue with the PYNQ experience, connect to JupyterLab via a web browser using this url: 10.8.3.232:9090/lab or k
ria:9090/lab - The password is xilinx
```

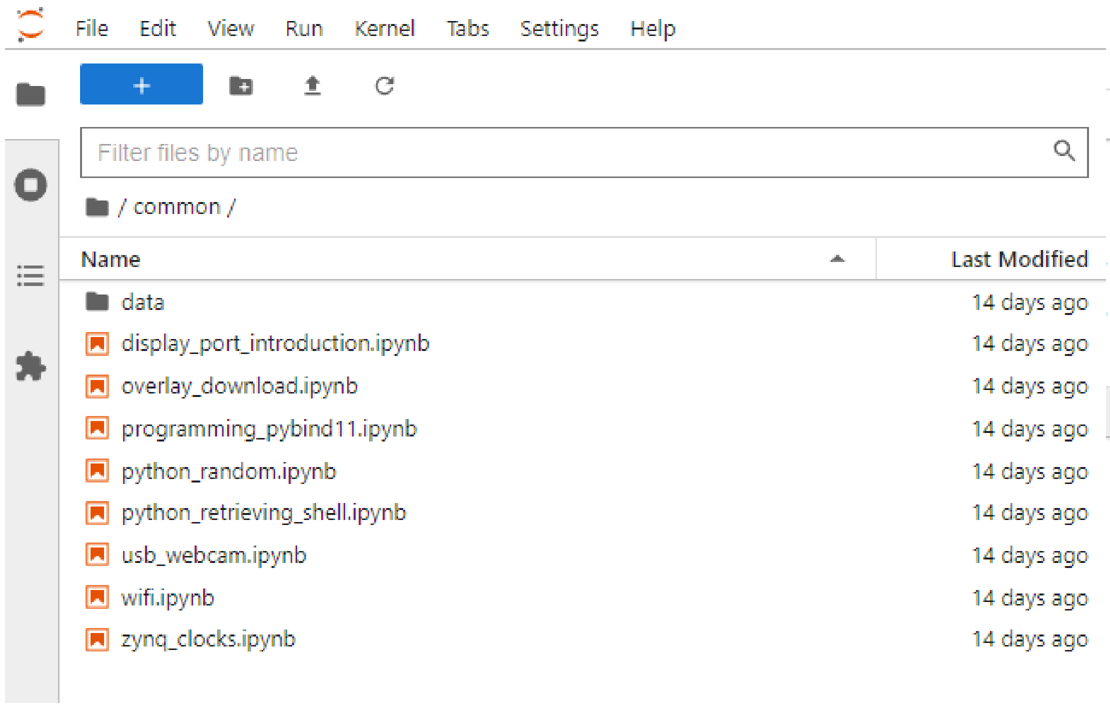
On your computer, enter the provided URL in yellow, e.g., 10.8.3.232:9090/lab, and enter the password "xilinx." You should then see:



Name	Last Modified
common	an hour ago
getting_started	an hour ago
kd240_notebooks	an hour ago
Welcome to Pynq.ipynb	a year ago

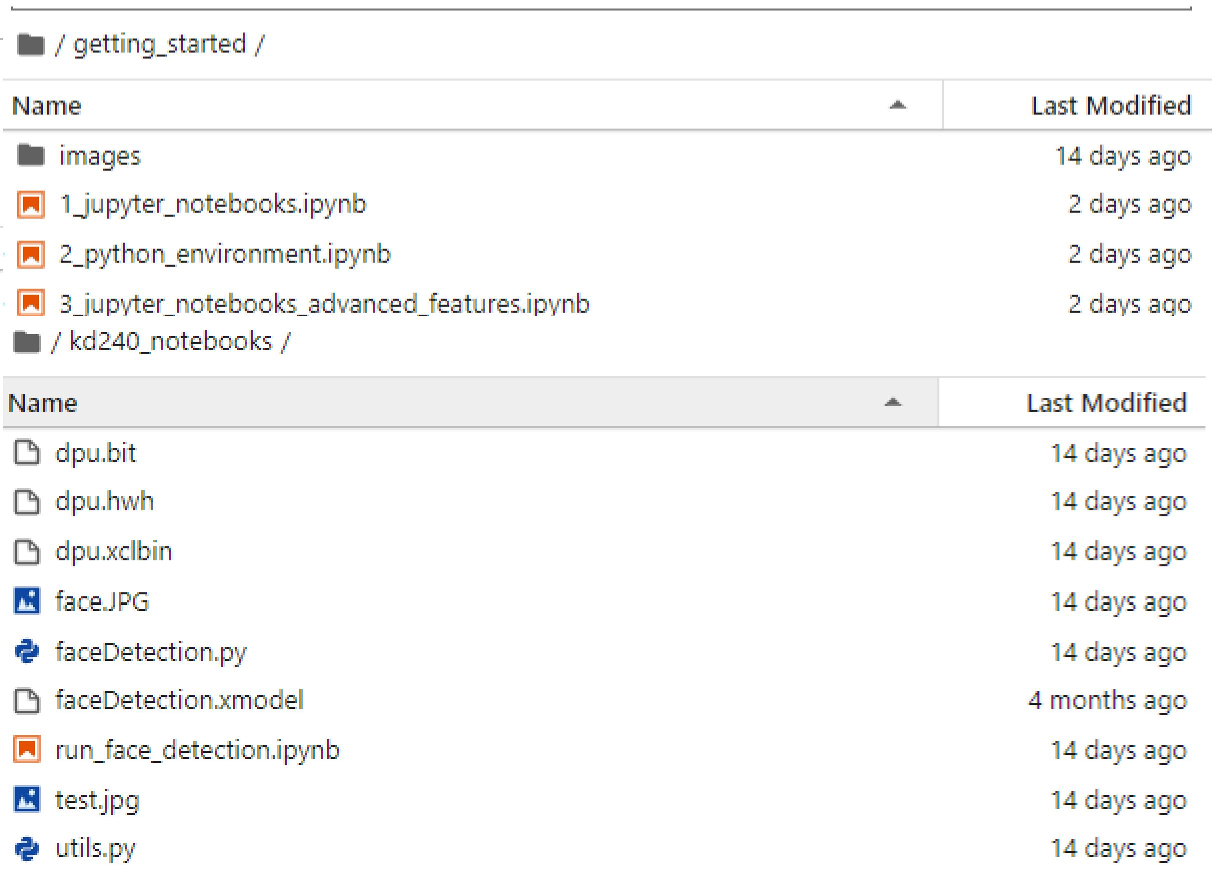
Example Notebooks

PYNQ uses the Jupyter Notebook environment to provide examples and documentation.



/ common /

Name	Last Modified
data	14 days ago
display_port_introduction.ipynb	14 days ago
overlay_download.ipynb	14 days ago
programming_pybind11.ipynb	14 days ago
python_random.ipynb	14 days ago
python_retrieving_shell.ipynb	14 days ago
usb_webcam.ipynb	14 days ago
wifi.ipynb	14 days ago
zynq_clocks.ipynb	14 days ago



/ getting_started /

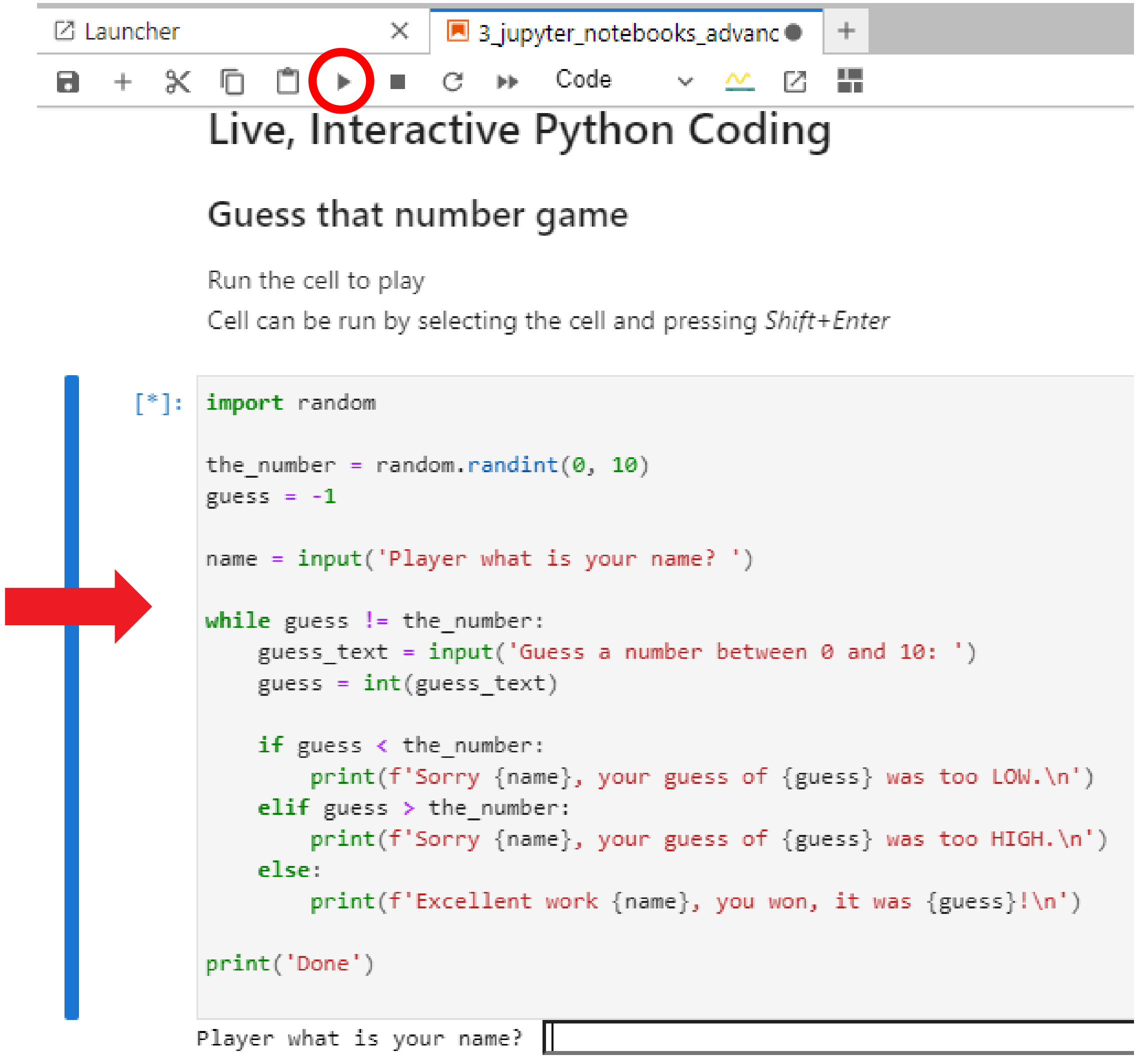
Name	Last Modified
images	14 days ago
1_jupyter_notebooks.ipynb	2 days ago
2_python_environment.ipynb	2 days ago
3_jupyter_notebooks_advanced_features.ipynb	2 days ago

/ kd240_notebooks /

Name	Last Modified
dpu.bit	14 days ago
dpu.hwh	14 days ago
dpu.xclbin	14 days ago
face.JPG	14 days ago
faceDetection.py	14 days ago
faceDetection.xmodel	4 months ago
run_face_detection.ipynb	14 days ago
test.jpg	14 days ago
utils.py	14 days ago

Start with a simple example

Click on the field and press "Run."



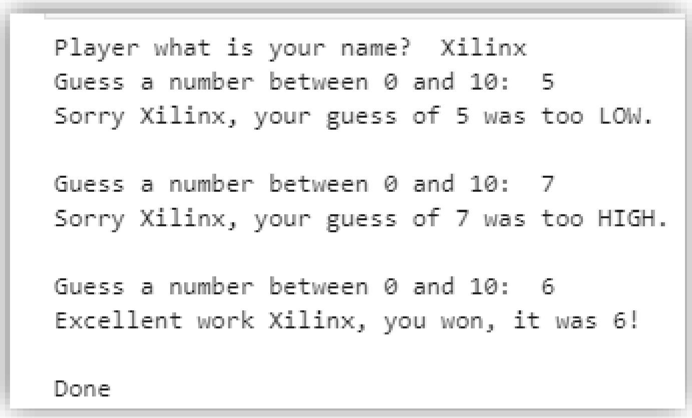
Live, Interactive Python Coding

Guess that number game

Run the cell to play

Cell can be run by selecting the cell and pressing *Shift+Enter*

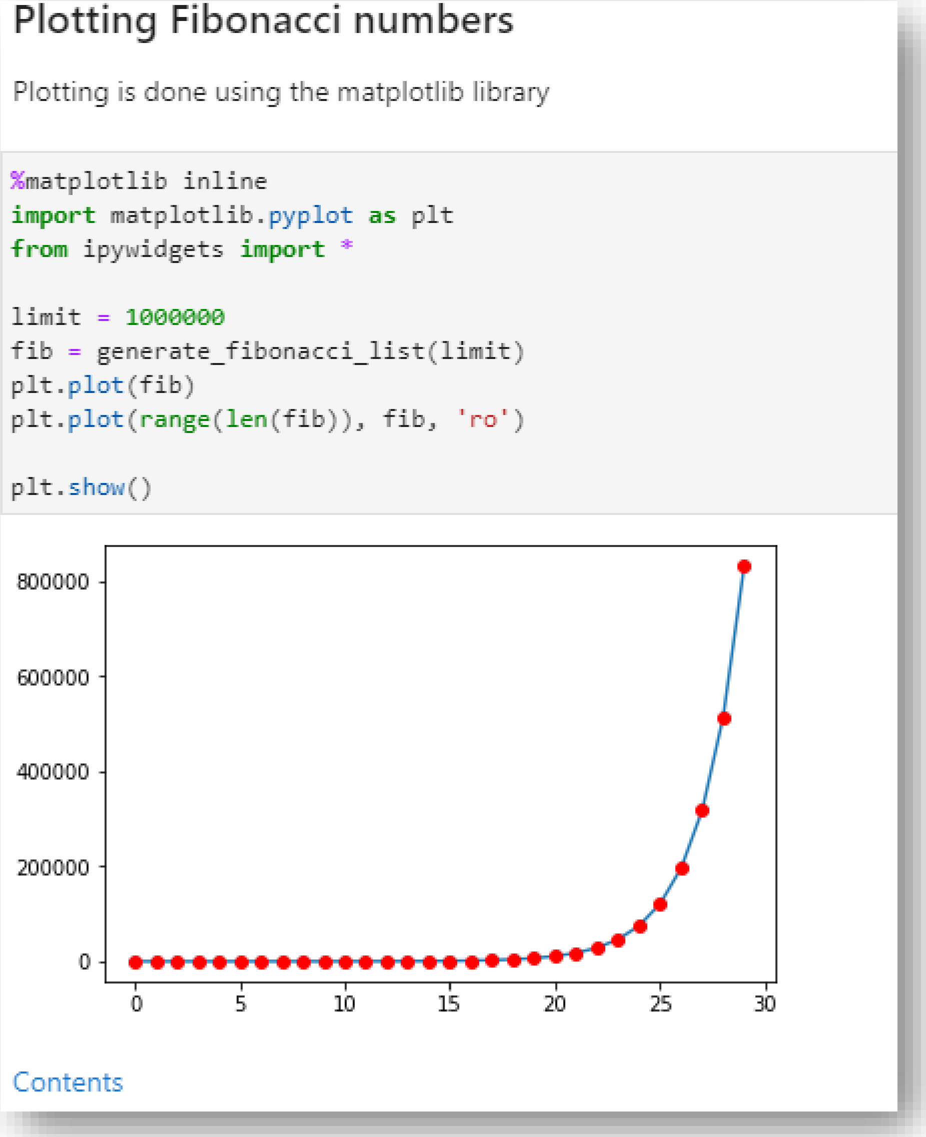
```
[*]: import random

     the_number = random.randint(0, 10)
     guess = -1

     name = input('Player what is your name? ')

     while guess != the_number:
         guess_text = input('Guess a number between 0 and 10: ')
         guess = int(guess_text)

         if guess < the_number:
             print(f'Sorry {name}, your guess of {guess} was too LOW.\n')
         elif guess > the_number:
             print(f'Sorry {name}, your guess of {guess} was too HIGH.\n')
         else:
             print(f'Excellent work {name}, you won, it was {guess}!\n')

     print('Done')

Player what is your name?
```



```
Player what is your name?  Xilinx
Guess a number between 0 and 10:  5
Sorry Xilinx, your guess of 5 was too LOW.

Guess a number between 0 and 10:  7
Sorry Xilinx, your guess of 7 was too HIGH.

Guess a number between 0 and 10:  6
Excellent work Xilinx, you won, it was 6!

Done
```

Matplotlib Library

Matplotlib library is already installed by default; there is no need for additional installation.

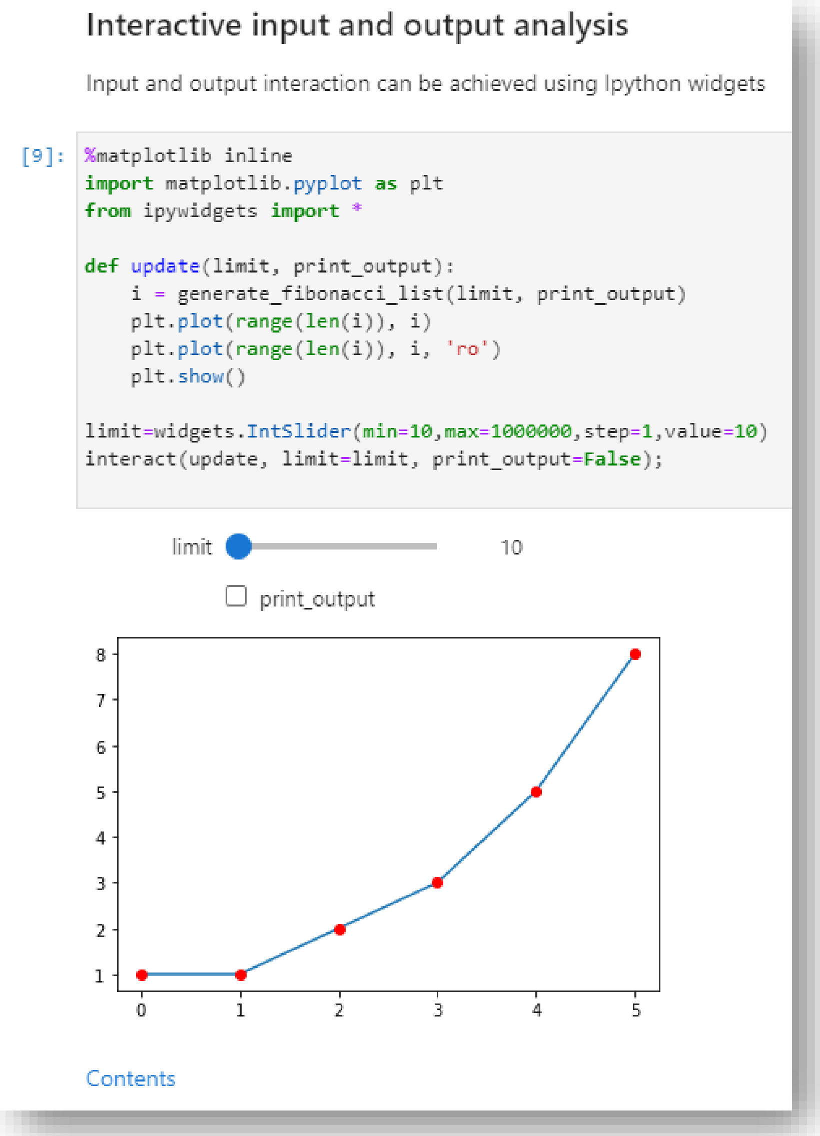


Plotting Fibonacci numbers

Plotting is done using the matplotlib library

```
%matplotlib inline
import matplotlib.pyplot as plt
from ipywidgets import *

limit = 1000000
fib = generate_fibonacci_list(limit)
plt.plot(fib)
plt.plot(range(len(fib)), fib, 'ro')

plt.show()
```

Contents



Interactive input and output analysis

Input and output interaction can be achieved using Ipython widgets

```
[9]: %matplotlib inline
     import matplotlib.pyplot as plt
     from ipywidgets import *

     def update(limit, print_output):
         i = generate_fibonacci_list(limit, print_output)
         plt.plot(range(len(i)), i)
         plt.plot(range(len(i)), i, 'ro')
         plt.show()

     limit=widgets.IntSlider(min=10,max=1000000,step=1,value=10)
     interact(update, limit=limit, print_output=False);
```

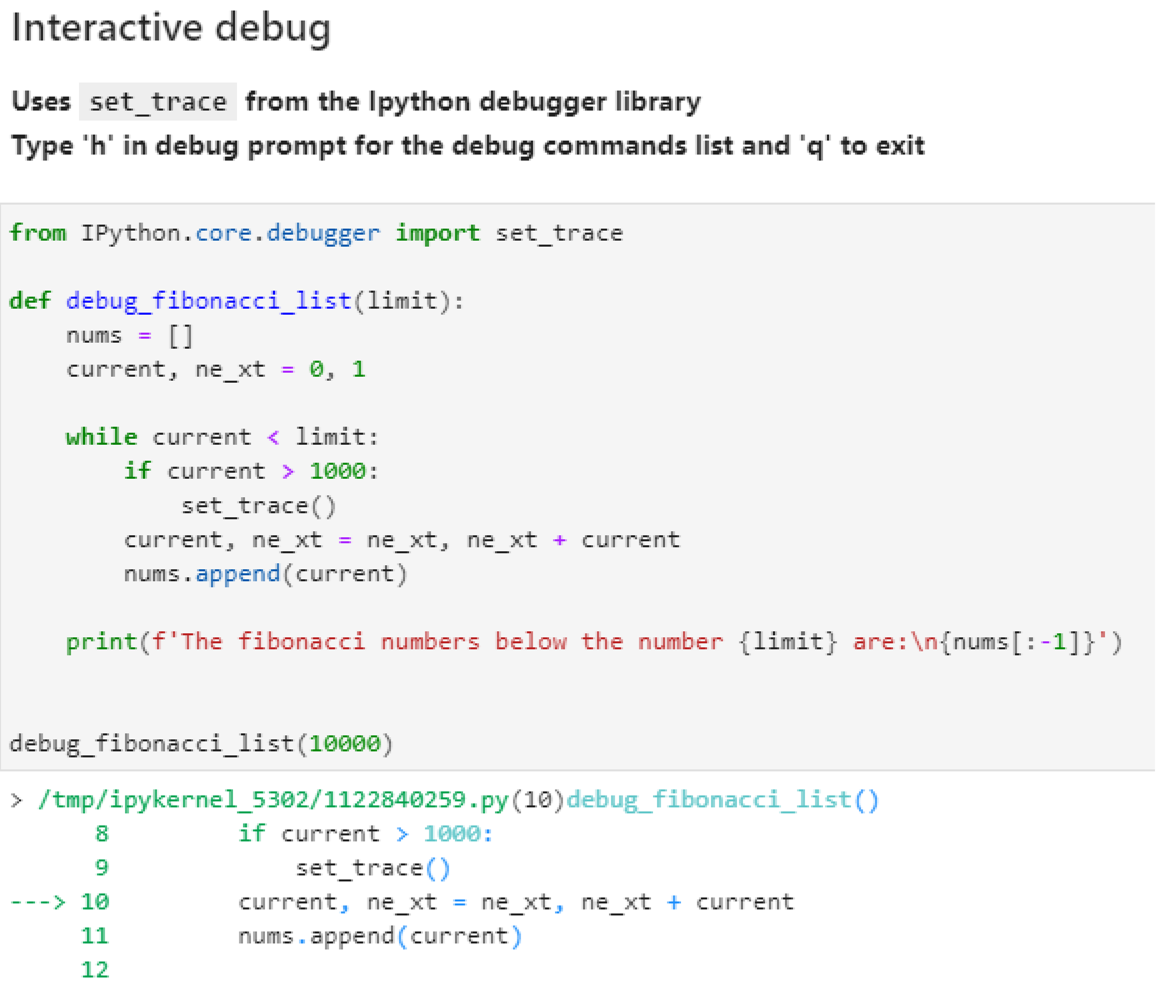
limit 10

□ print_output

Contents

Ipython.Core.debugger

Uses set_trace to create a breakpoint in your function.



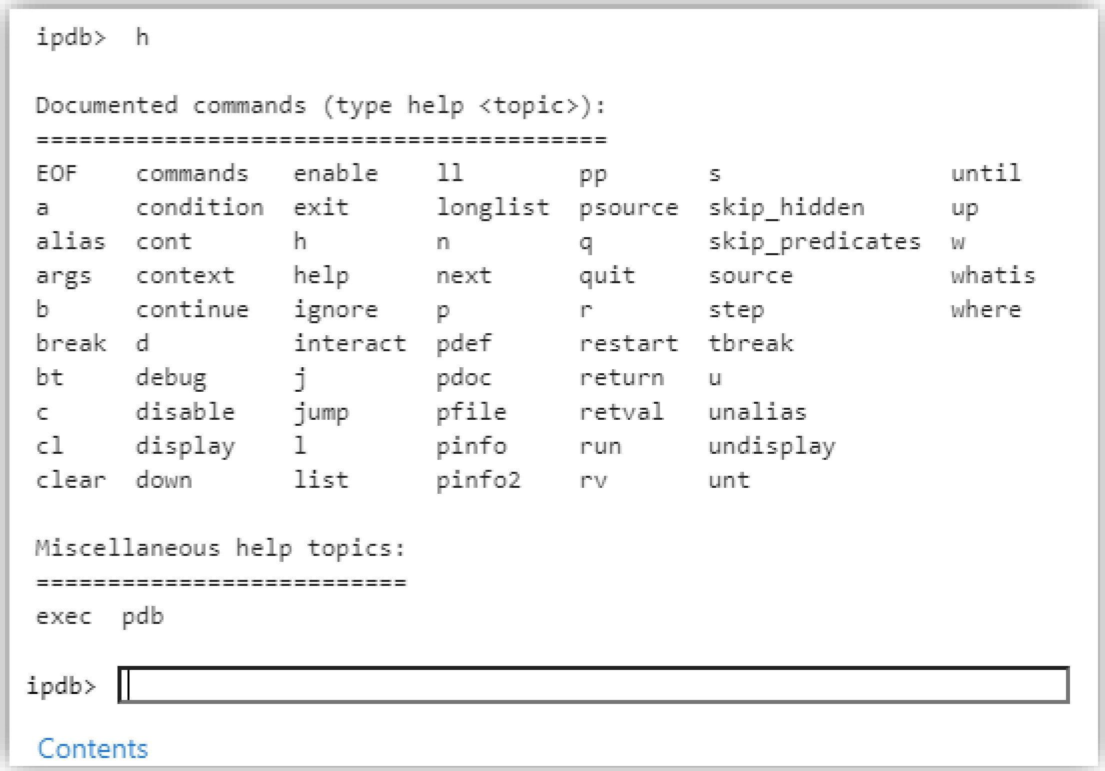
Interactive debug

Uses `set_trace` from the Ipython debugger library

Type 'h' in debug prompt for the debug commands list and 'q' to exit

```
from IPython.core.debugger import set_trace

def debug_fibonacci_list(limit):
    nums = []
    current, ne_xt = 0, 1

    while current < limit:
        if current > 1000:
            set_trace()
        current, ne_xt = ne_xt, ne_xt + current
        nums.append(current)

    print(f'The fibonacci numbers below the number {limit} are:\n{nums[:-1]}')


debug_fibonacci_list(10000)
```

```
> /tmp/ipykernel_5302/1122840259.py(10)debug_fibonacci_list()
      8         if current > 1000:
      9             set_trace()
---> 10         current, ne_xt = ne_xt, ne_xt + current
     11         nums.append(current)
     12
```



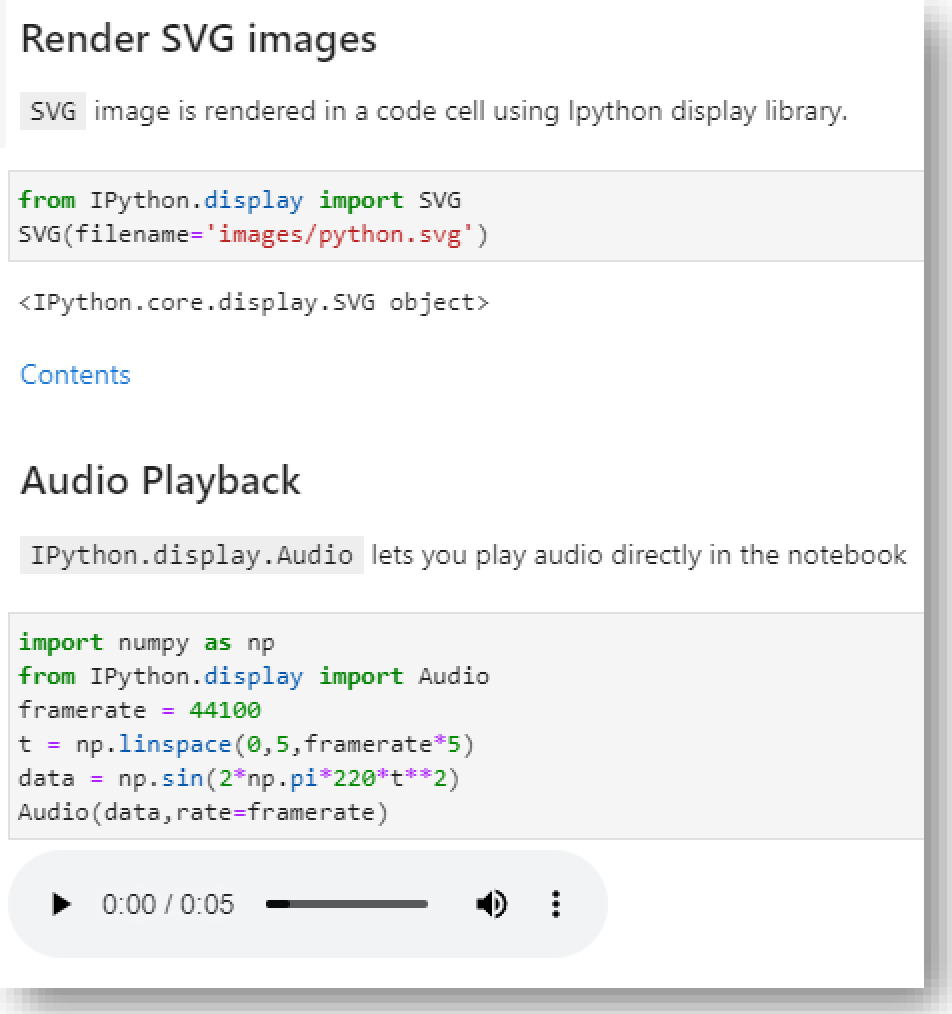
```
ipdb>  h

Documented commands (type help <topic>):
========================================
EOF    commands   enable    ll        pp       s                until
a      condition  exit      longlist  psource  skip_hidden      up
alias  cont       h         n         q        skip_predicates  w
args   context    help      next      quit     source           whatis
b      continue   ignore    p         r        step             where
break  d          interact  pdef      restart  tbreak
bt     debug      j         pdoc      return   u
c      disable    jump      pfile     retval   unalias
cl     display    l         pinfo     run      undisplay
clear  down       list      pinfo2    rv       unt

Miscellaneous help topics:
==========================
exec  pdb

ipdb>
```

Contents

AMD together we advance_

Ipython.Core.Display

Audio Playback & Add Video



Render SVG images

`SVG` image is rendered in a code cell using Ipython display library.

```
from IPython.display import SVG
SVG(filename='images/python.svg')
```

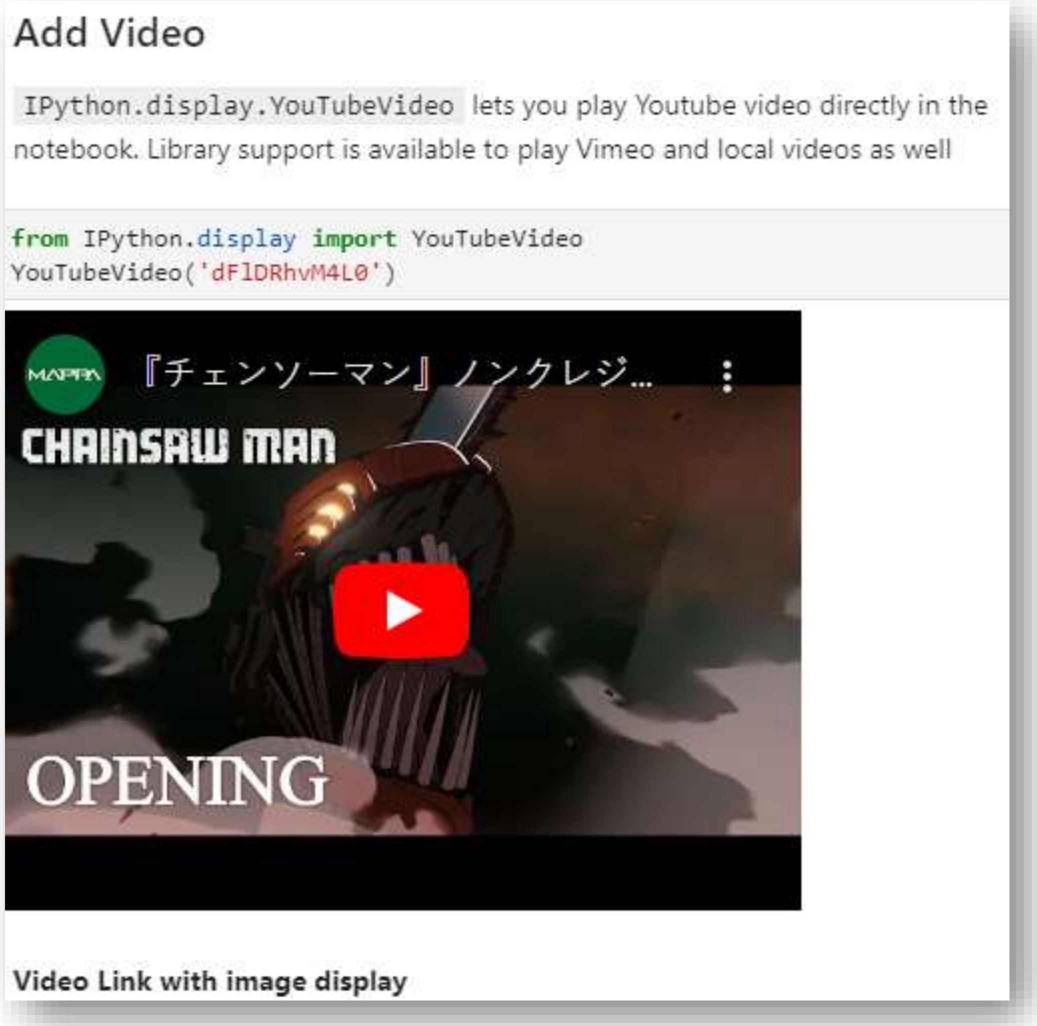
```
<IPython.core.display.SVG object>
```

Contents

Audio Playback

`IPython.display.Audio` lets you play audio directly in the notebook

```
import numpy as np
from IPython.display import Audio
framerate = 44100
t = np.linspace(0,5,framerate*5)
data = np.sin(2*np.pi*220*t**2)
Audio(data,rate=framerate)
```



Add Video

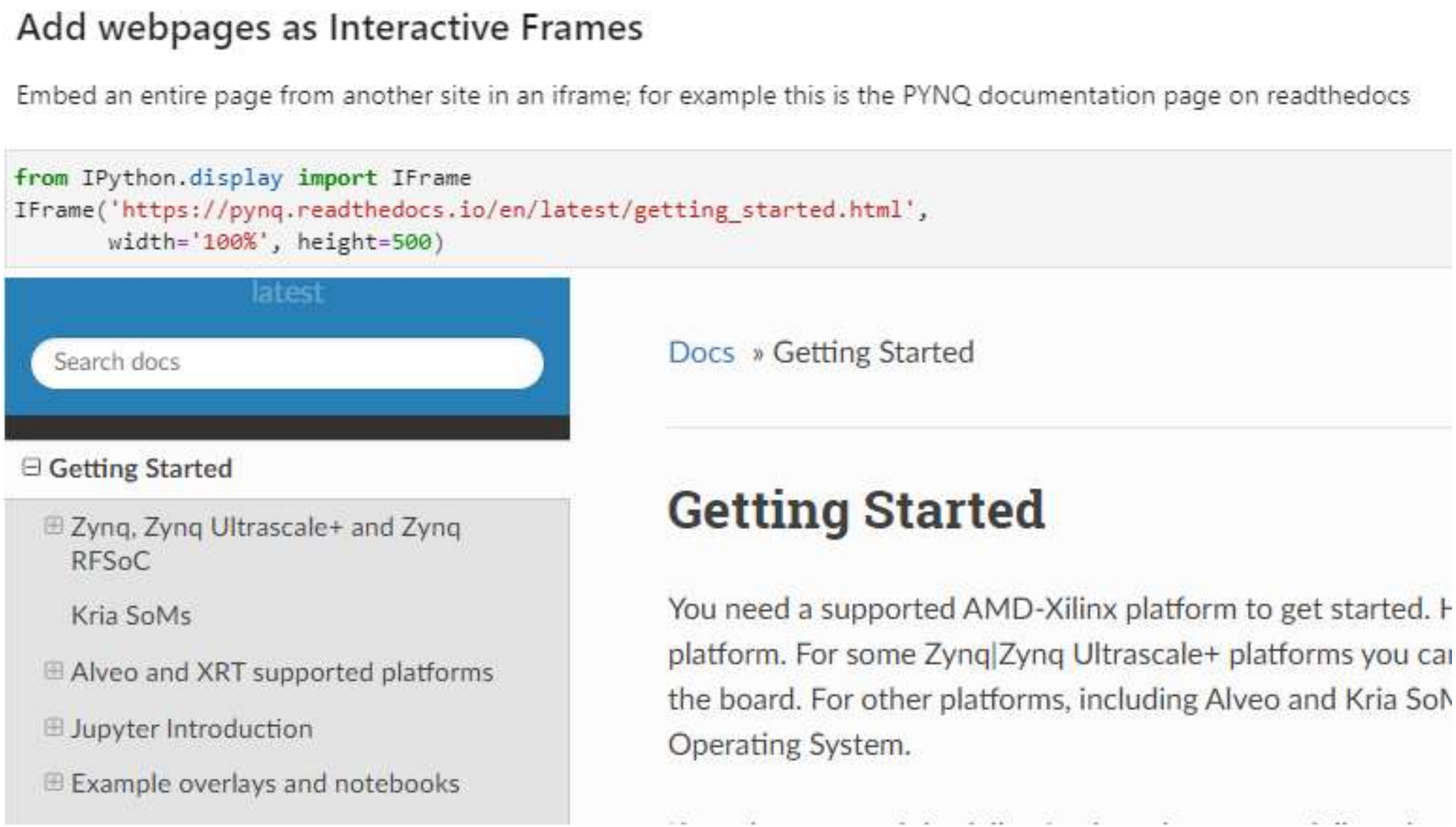
`IPython.display.YouTubeVideo` lets you play Youtube video directly in the notebook. Library support is available to play Vimeo and local videos as well

```
from IPython.display import YouTubeVideo
YouTubeVideo('dFlDRhvM4L0')
```

Video Link with image display

Ipython.Core.Display

Add webpages as Interactive Frames



Add webpages as Interactive Frames

Embed an entire page from another site in an iframe; for example this is the PYNQ documentation page on readthedocs

```
from IPython.display import IFrame
IFrame('https://pynq.readthedocs.io/en/latest/getting_started.html',
       width='100%', height=500)
```

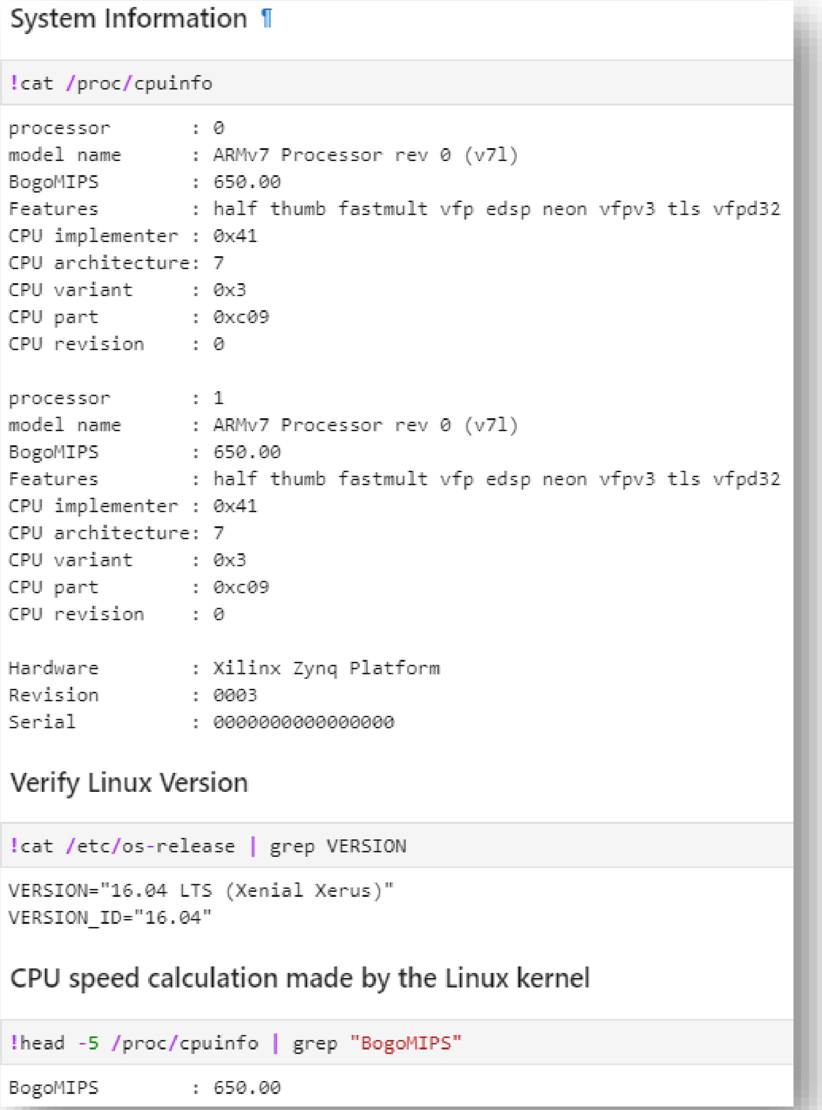
latest
Search docs
⊟ Getting Started
⊞ Zynq, Zynq Ultrascale+ and Zynq RFSoC
Kria SoMs
⊞ Alveo and XRT supported platforms
⊞ Jupyter Introduction
⊞ Example overlays and notebooks
Docs » Getting Started

Getting Started

You need a supported AMD-Xilinx platform to get started. H
platform. For some Zynq|Zynq Ultrascale+ platforms you ca
the board. For other platforms, including Alveo and Kria So
Operating System.

Notebooks are not just for Python

Access to linux shell commands



System Information ¶

```
!cat /proc/cpuinfo
```

```
processor       : 0
model name      : ARMv7 Processor rev 0 (v7l)
BogoMIPS        : 650.00
Features        : half thumb fastmult vfp edsp neon vfpv3 tls vfpd32
CPU implementer : 0x41
CPU architecture: 7
CPU variant     : 0x3
CPU part        : 0xc09
CPU revision    : 0

processor       : 1
model name      : ARMv7 Processor rev 0 (v7l)
BogoMIPS        : 650.00
Features        : half thumb fastmult vfp edsp neon vfpv3 tls vfpd32
CPU implementer : 0x41
CPU architecture: 7
CPU variant     : 0x3
CPU part        : 0xc09
CPU revision    : 0

Hardware        : Xilinx Zynq Platform
Revision        : 0003
Serial          : 0000000000000000
```

Verify Linux Version

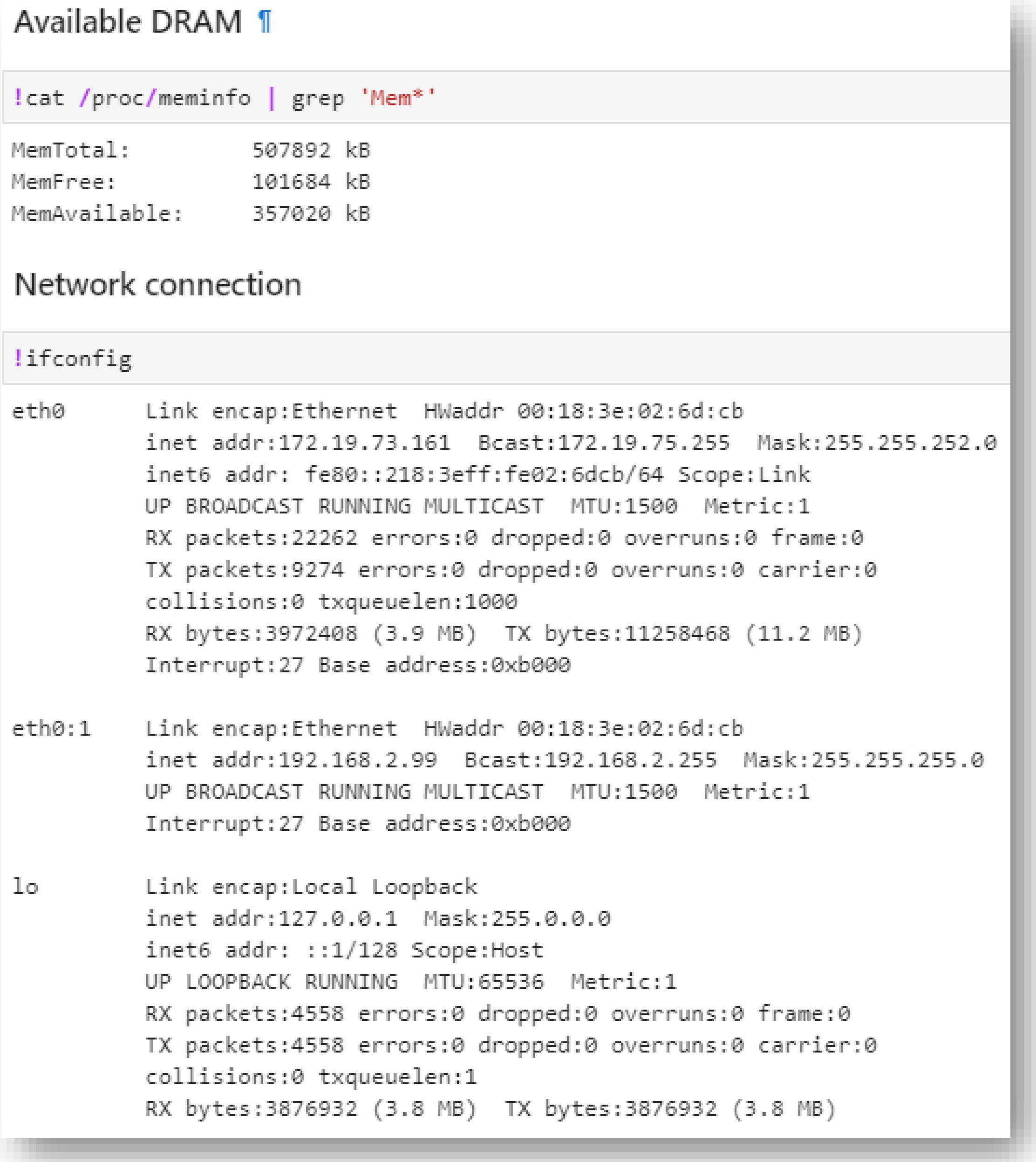
```
!cat /etc/os-release | grep VERSION
```

```
VERSION="16.04 LTS (Xenial Xerus)"
VERSION_ID="16.04"
```

CPU speed calculation made by the Linux kernel

```
!head -5 /proc/cpuinfo | grep "BogoMIPS"
```

```
BogoMIPS        : 650.00
```



Available DRAM ¶

```
!cat /proc/meminfo | grep 'Mem*'
```

```
MemTotal:         507892 kB
MemFree:          101684 kB
MemAvailable:     357020 kB
```

Network connection

```
!ifconfig
```

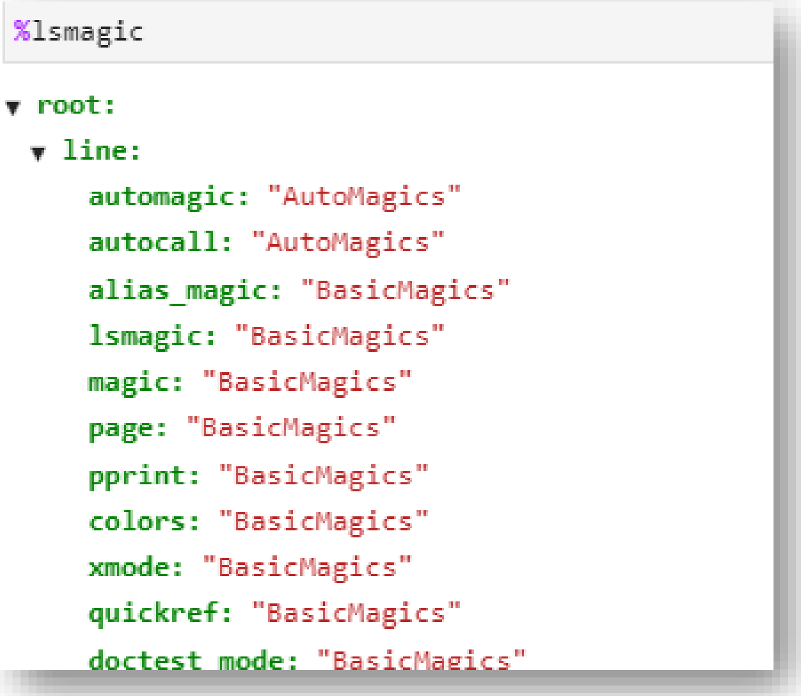
```
eth0      Link encap:Ethernet  HWaddr 00:18:3e:02:6d:cb  
          inet addr:172.19.73.161  Bcast:172.19.75.255  Mask:255.255.252.0
          inet6 addr: fe80::218:3eff:fe02:6dcb/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:22262 errors:0 dropped:0 overruns:0 frame:0
          TX packets:9274 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000 
          RX bytes:3972408 (3.9 MB)  TX bytes:11258468 (11.2 MB)
          Interrupt:27 Base address:0xb000 

eth0:1    Link encap:Ethernet  HWaddr 00:18:3e:02:6d:cb  
          inet addr:192.168.2.99  Bcast:192.168.2.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:27 Base address:0xb000 

lo        Link encap:Local Loopback  
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:4558 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4558 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1 
          RX bytes:3876932 (3.8 MB)  TX bytes:3876932 (3.8 MB)
```

Magic

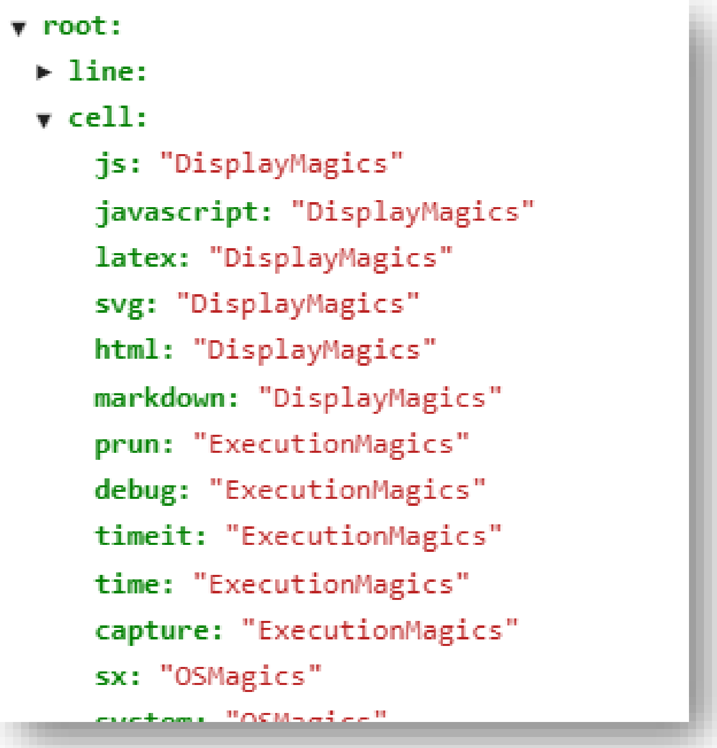
IPython has a set of predefined ‘magic functions’ that you can call with a command line style syntax.

To learn more about the IPython magics, simple type %magic in a separate cell.

Below is a list of available magics.

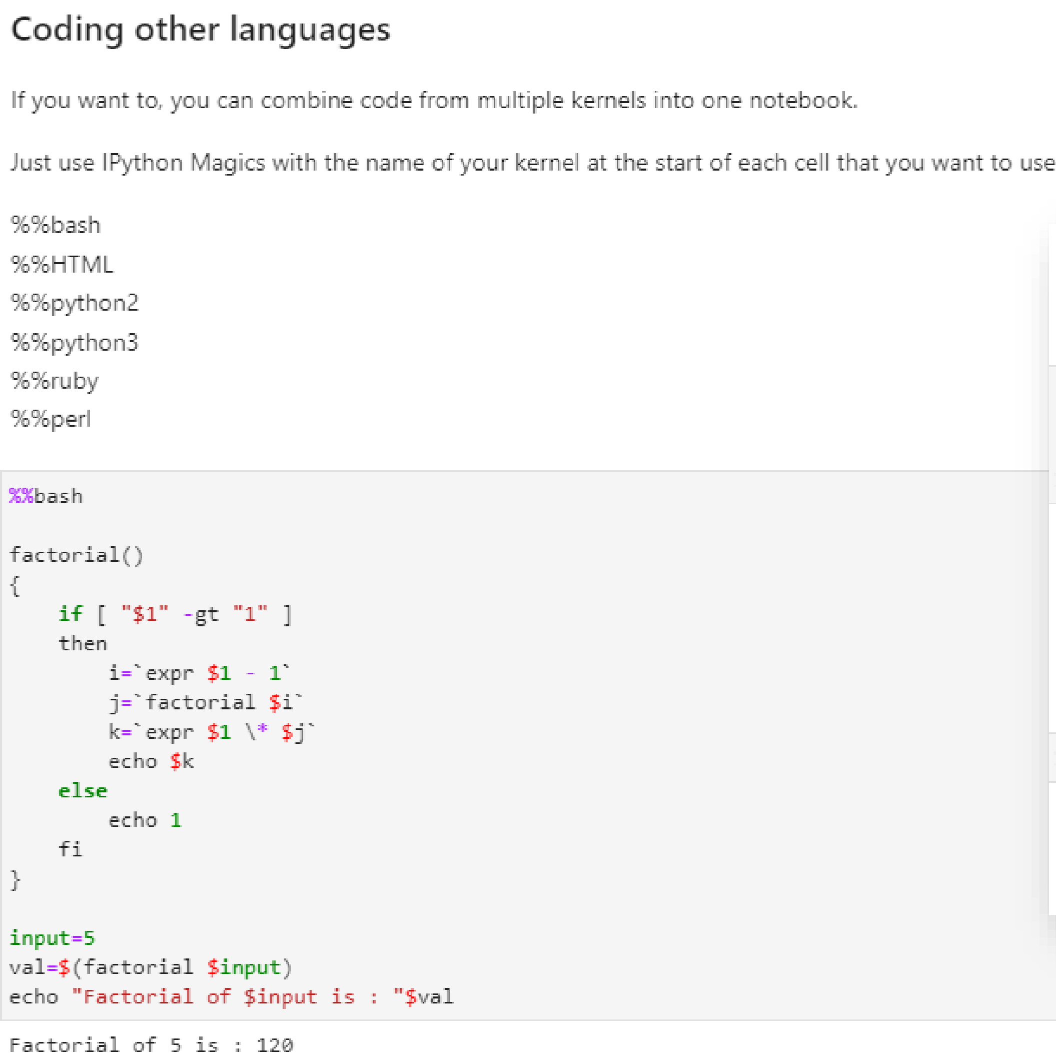


```
%lsmagic

▼ root:
  ▼ line:
      automagic: "AutoMagics"
      autocall: "AutoMagics"
      alias_magic: "BasicMagics"
      lsmagic: "BasicMagics"
      magic: "BasicMagics"
      page: "BasicMagics"
      pprint: "BasicMagics"
      colors: "BasicMagics"
      xmode: "BasicMagics"
      quickref: "BasicMagics"
      doctest_mode: "BasicMagics"
```



```
▼ root:
  ▶ line:
  ▼ cell:
      js: "DisplayMagics"
      javascript: "DisplayMagics"
      latex: "DisplayMagics"
      svg: "DisplayMagics"
      html: "DisplayMagics"
      markdown: "DisplayMagics"
      prun: "ExecutionMagics"
      debug: "ExecutionMagics"
      timeit: "ExecutionMagics"
      time: "ExecutionMagics"
      capture: "ExecutionMagics"
      sx: "OSMagics"
      system: "OSMagics"
```

Magic Function Example



Coding other languages

If you want to, you can combine code from multiple kernels into one notebook.

Just use IPython Magics with the name of your kernel at the start of each cell that you want to use that Kernel for:

%%bash

%%HTML

%%python2

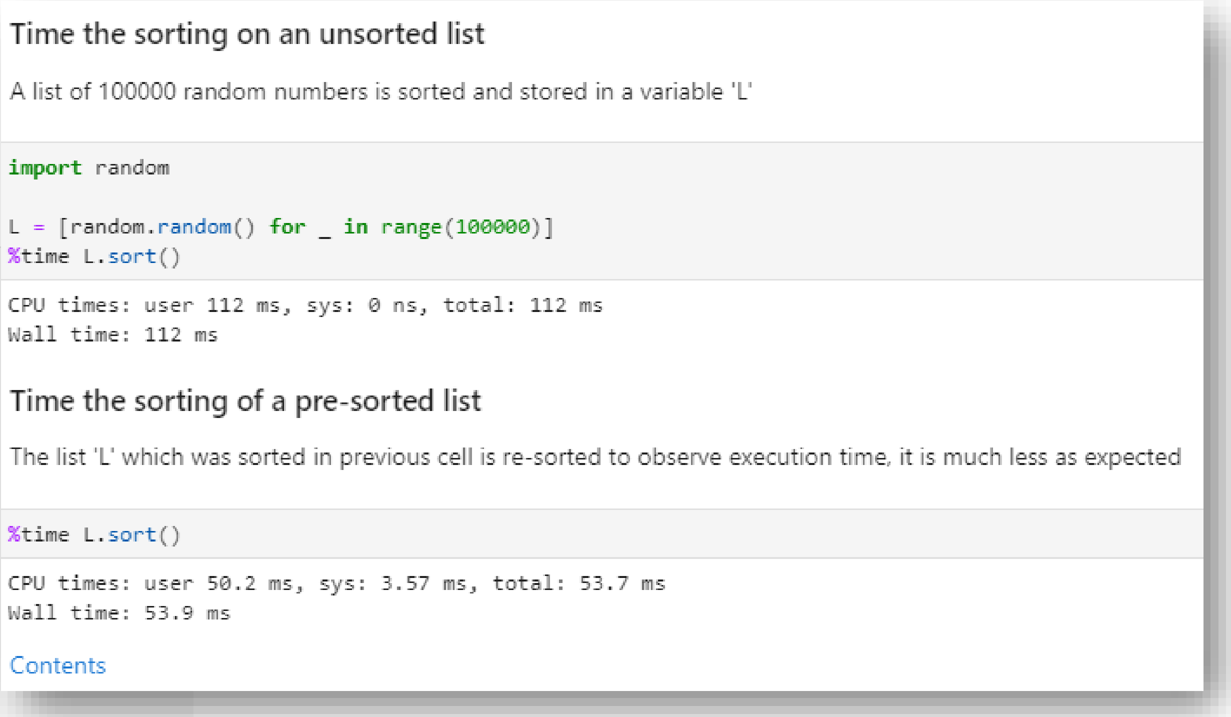
%%python3

%%ruby

%%perl

```
%%bash

factorial()
{
    if [ "$1" -gt "1" ]
    then
        i=`expr $1 - 1`
        j=`factorial $i`
        k=`expr $1 \* $j`
        echo $k
    else
        echo 1
    fi
}

input=5
val=$(factorial $input)
echo "Factorial of $input is : "$val
```

```
Factorial of 5 is : 120
```



Time the sorting on an unsorted list

A list of 100000 random numbers is sorted and stored in a variable 'L'

```
import random

L = [random.random() for _ in range(100000)]
%time L.sort()
```

```
CPU times: user 112 ms, sys: 0 ns, total: 112 ms
Wall time: 112 ms
```

Time the sorting of a pre-sorted list

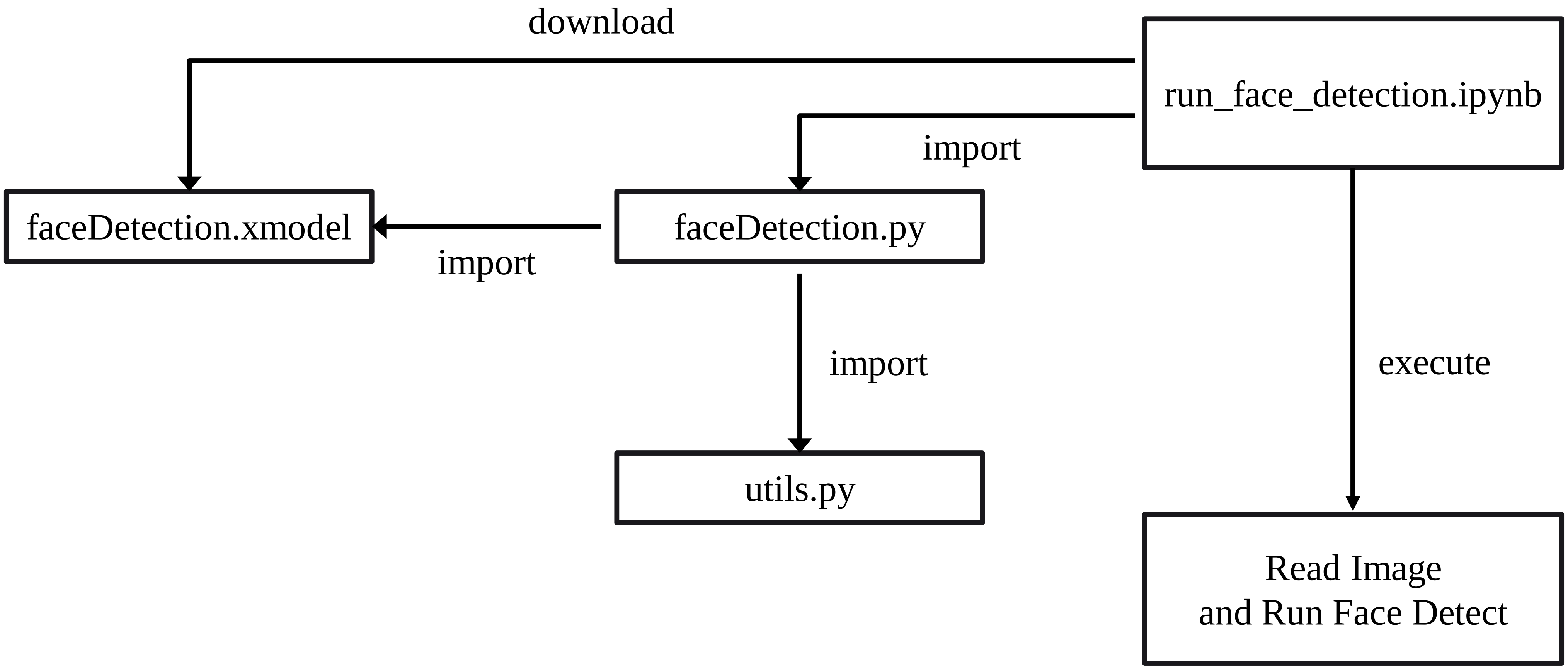
The list 'L' which was sorted in previous cell is re-sorted to observe execution time, it is much less as expected

```
%time L.sort()
```

```
CPU times: user 50.2 ms, sys: 3.57 ms, total: 53.7 ms
Wall time: 53.9 ms
```

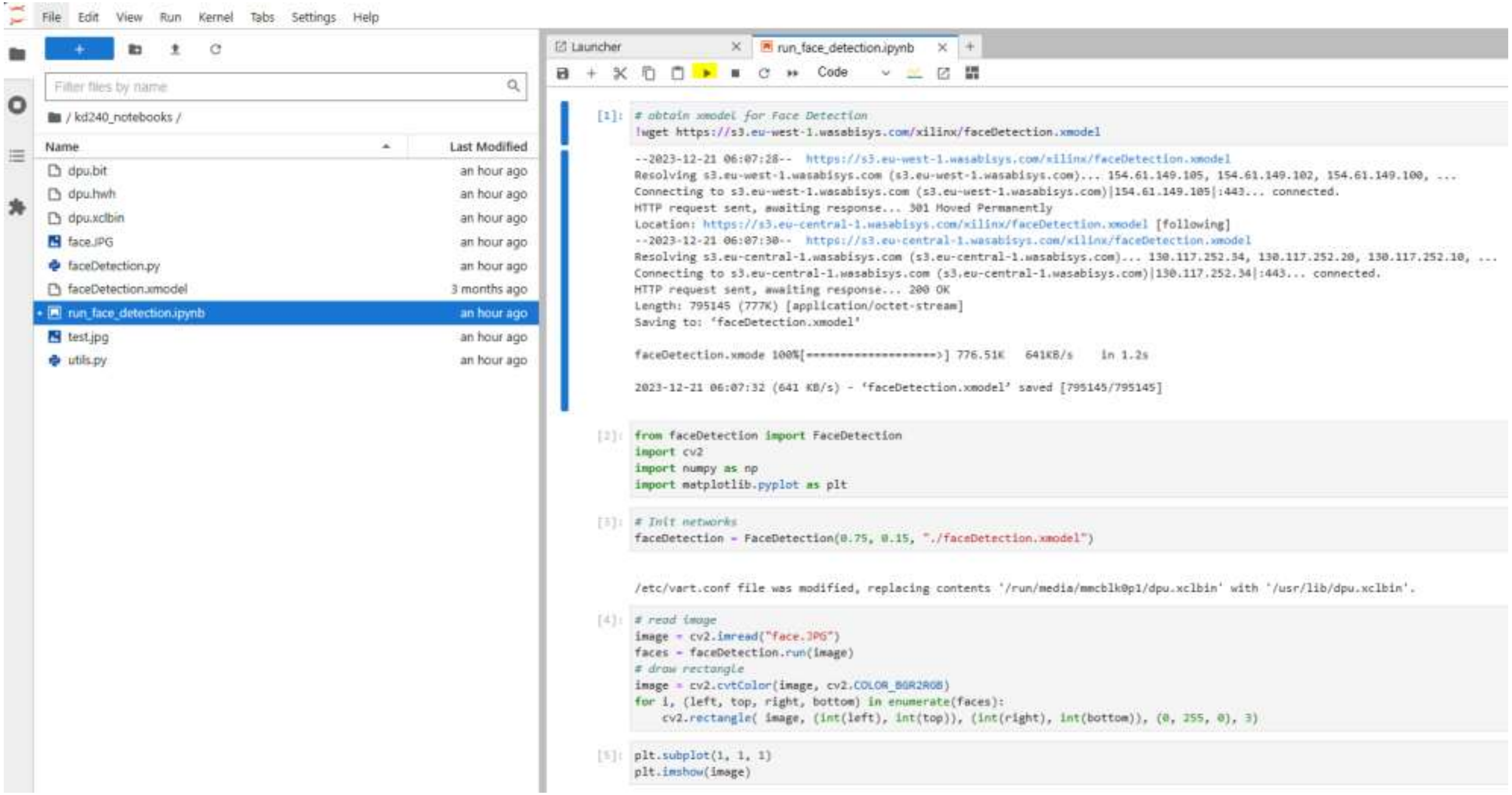
Contents

APPENDIX: Code Flow



Start with a simple example

10. 點開 kd240_notebooks，再點開 run_face_detection.ipynb，可以發現會去呼叫 faceDetection.py，並且在 faceDetection.py 中會再去呼叫 utils.py，可以點選圖中黃色撥放鍵逐一執行程式



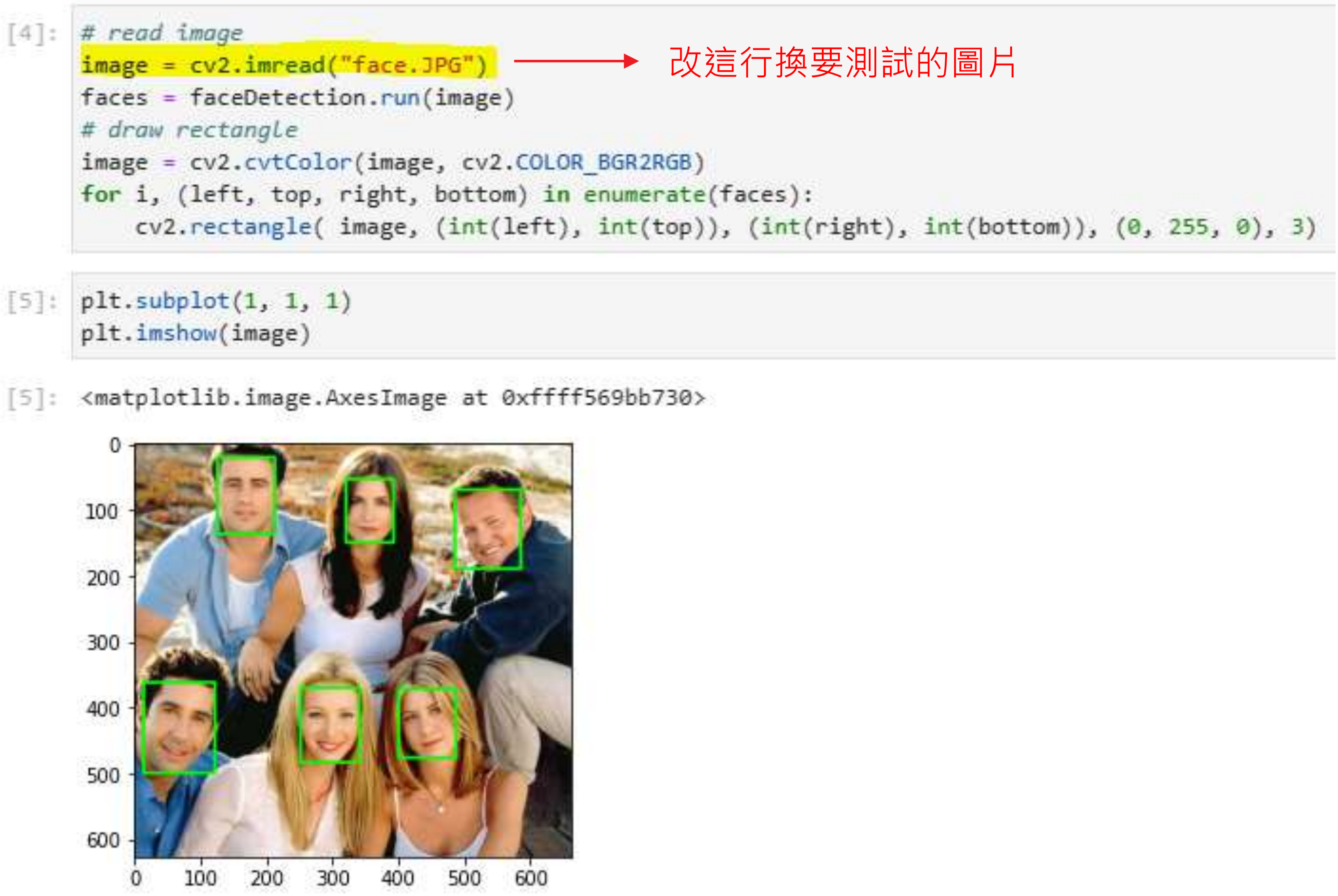
/ kd240_notebooks /

Name	Last Modified
dpu.bit	an hour ago
dpu.hwh	an hour ago
dpu.xclbin	an hour ago
face.JPG	an hour ago
faceDetection.py	an hour ago
faceDetection.xmodel	3 months ago
run_face_detection.ipynb	an hour ago
test.jpg	an hour ago
utils.py	an hour ago

```
[1]: # obtain xmodel for Face Detection
     !wget https://s3.eu-west-1.wasabisys.com/xilinx/faceDetection.xmodel

     --2023-12-21 06:07:28--  https://s3.eu-west-1.wasabisys.com/xilinx/faceDetection.xmodel
     Resolving s3.eu-west-1.wasabisys.com (s3.eu-west-1.wasabisys.com)... 154.61.149.105, 154.61.149.102, 154.61.149.100, ...
     Connecting to s3.eu-west-1.wasabisys.com (s3.eu-west-1.wasabisys.com)|154.61.149.105|:443... connected.
     HTTP request sent, awaiting response... 301 Moved Permanently
     Location: https://s3.eu-central-1.wasabisys.com/xilinx/faceDetection.xmodel [following]
     --2023-12-21 06:07:30--  https://s3.eu-central-1.wasabisys.com/xilinx/faceDetection.xmodel
     Resolving s3.eu-central-1.wasabisys.com (s3.eu-central-1.wasabisys.com)... 130.117.252.34, 130.117.252.20, 130.117.252.10, ...
     Connecting to s3.eu-central-1.wasabisys.com (s3.eu-central-1.wasabisys.com)|130.117.252.34|:443... connected.
     HTTP request sent, awaiting response... 200 OK
     Length: 795145 (777K) [application/octet-stream]
     Saving to: 'faceDetection.xmodel'

     faceDetection.xmode 100%[===================>] 776.51K   641KB/s    in 1.2s

     2023-12-21 06:07:32 (641 KB/s) - 'faceDetection.xmodel' saved [795145/795145]

[2]: from faceDetection import FaceDetection
     import cv2
     import numpy as np
     import matplotlib.pyplot as plt

[3]: # Init networks
     faceDetection = FaceDetection(0.75, 0.15, "./faceDetection.xmodel")

     /etc/vart.conf file was modified, replacing contents '/run/media/mmcblk0p1/dpu.xclbin' with '/usr/lib/dpu.xclbin'.

[4]: # read image
     image = cv2.imread("face.JPG")
     faces = faceDetection.run(image)
     # draw rectangle
     image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
     for i, (left, top, right, bottom) in enumerate(faces):
         cv2.rectangle( image, (int(left), int(top)), (int(right), int(bottom)), (0, 255, 0), 3)

[5]: plt.subplot(1, 1, 1)
     plt.imshow(image)
```

KD240 Face Detect

11. 最後可以發現結果會顯示在畫面上，並且可以自己抓圖片來做測試



```
[4]: # read image
     image = cv2.imread("face.JPG")
     faces = faceDetection.run(image)
     # draw rectangle
     image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
     for i, (left, top, right, bottom) in enumerate(faces):
         cv2.rectangle( image, (int(left), int(top)), (int(right), int(bottom)), (0, 255, 0), 3)
```

改這行換要測試的圖片

```
[5]: plt.subplot(1, 1, 1)
     plt.imshow(image)
```

```
[5]: <matplotlib.image.AxesImage at 0xffff569bb730>
```

AMD