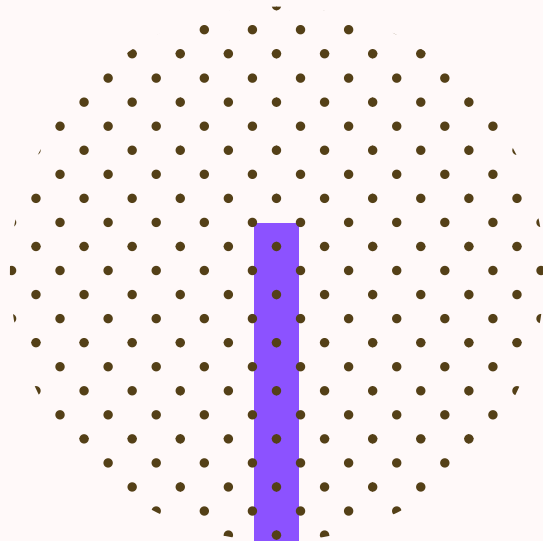


# INSTALLATION

BRONZE-QISKIT



## Welcome to QBronze-Qiskit!

This workshop consists of a collection of Jupyter notebooks

We use Python 3 (3.6+) as our programming language and we will be working on the library Qiskit!

This is a beginners guide to install Qiskit!

*Note: If you have already installed Qiskit on your system, you may skip this guide and move on to the Start notebook*

*If you have Jupyter notebook/lab already installed, you may skip to the Install Qiskit section for the installation part*

### Installing Anaconda®



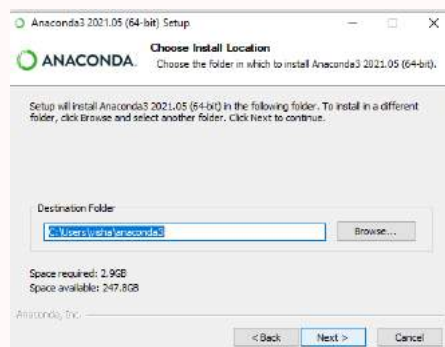
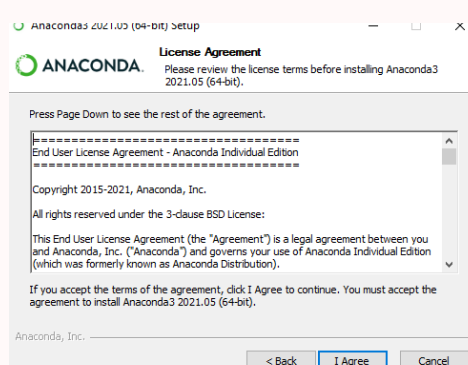
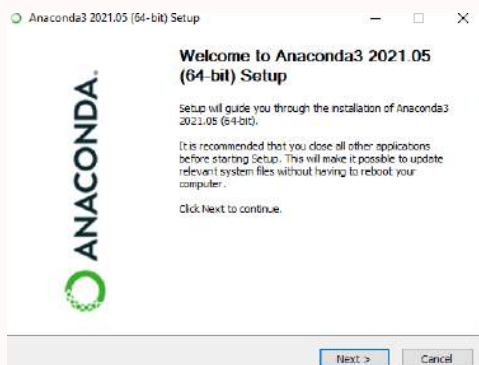
Installing Anaconda® will install the required Python compiler and libraries you need with one click! It will also make it easy to install Jupyter notebooks

[Click here to Download Anaconda](#)

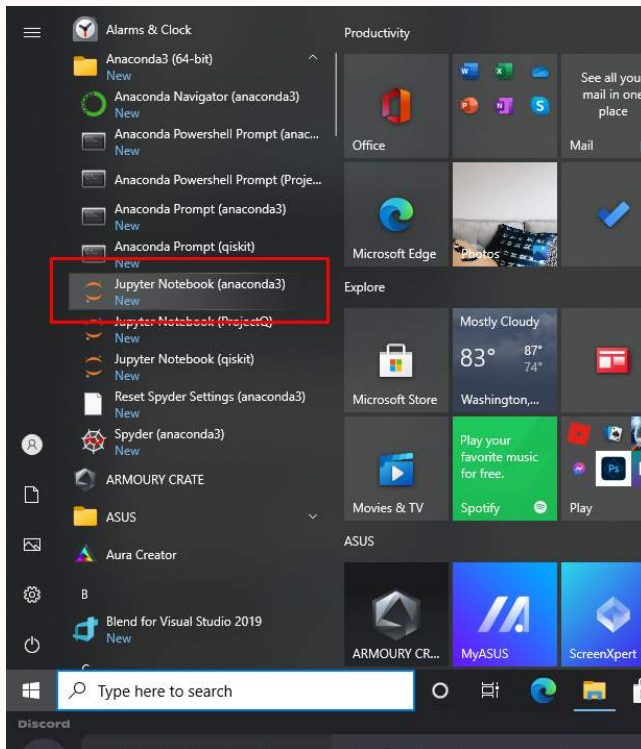
## Setting up Anaconda®



Select your preferred operating system and download the setup file. Run the setup file after downloading

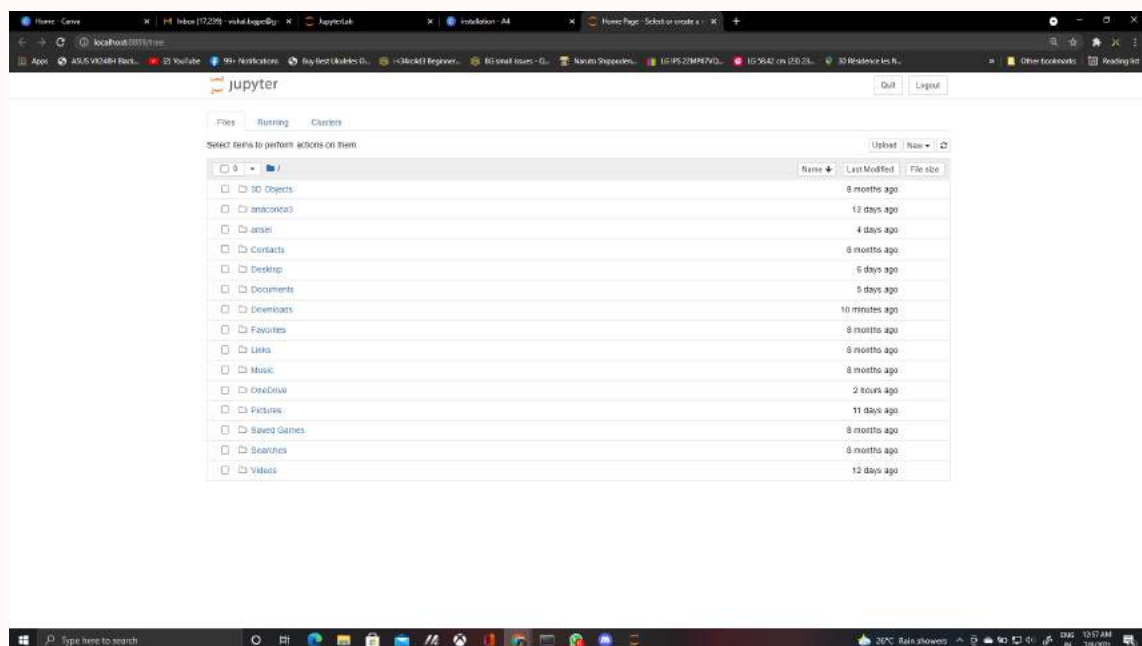


## Setting up Jupyter notebooks



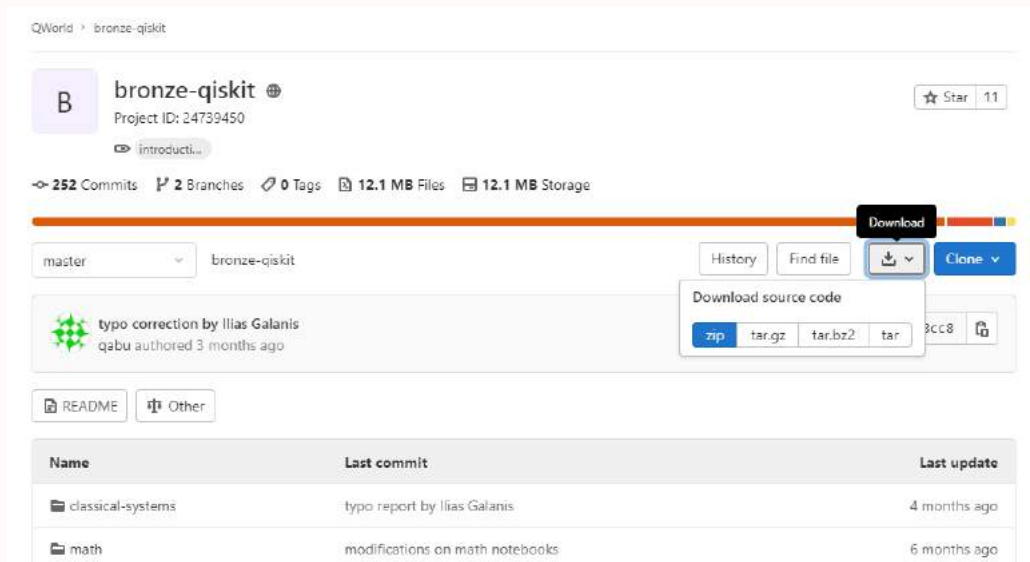
Run jupyter notebook from the start menu. A command prompt or powershell window should appear following an opening of a web browser page that looks like one below.

**Note:** Keep the command prompt window open while working.



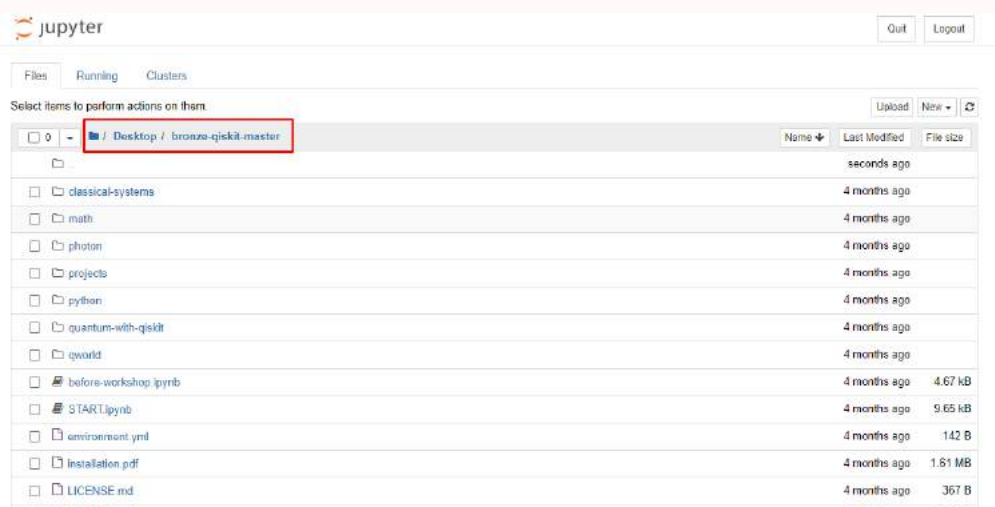
## Setting up QBronze files

Download the QBronze-Qiskit files from this [link](#)



Extract the files to one of the accessible directories e.g. Desktop, Documents, Downloads etc.

*We have extracted our files in the Desktop here and we can access it from the dashboard as shown*



# Installing Qiskit

1) Open the notebook "before-workshop.ipynb" from the dashboard and open "Qiskit installation and test"



2) When you run the first cell, you should get an error if Qiskit is not installed on your system

```
Check your system
Check your system, if Qiskit has already been installed:

In [1]: import qiskit
        versions = qiskit.__qiskit_version__
        print("The version of Qiskit is", versions['qiskit'])
        print()
        print("The version of each component:")
        for key in versions:
            print(key, "->", versions[key])

-----
AttributeError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_54728\3518098202.py in <module>
      1 import qiskit
----> 2 versions = qiskit.__qiskit_version__
      3 print("The version of Qiskit is", versions['qiskit'])
      4 print()
      5 print("The version of each component:")

AttributeError: module 'qiskit' has no attribute '__qiskit_version__'
```

## Installing Qiskit

### 3) Run this cell to initiate installation of Qiskit

```
Install qiskit

(If you are an experienced user, visit this link: https://qiskit.org/documentation/install.html)

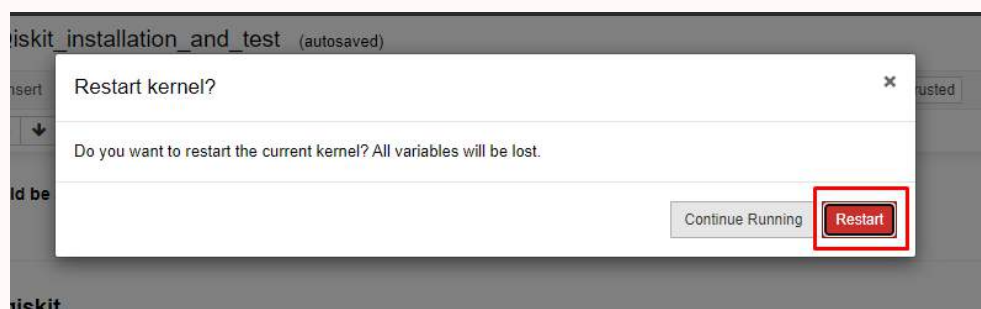
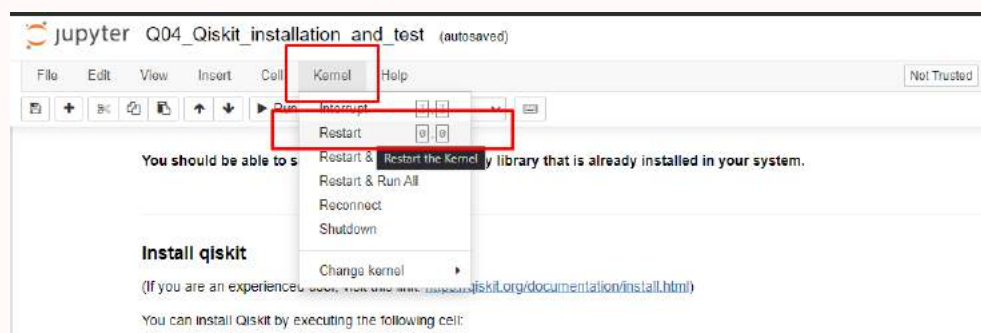
You can install Qiskit by executing the following cell:

In [2]: !pip install "qiskit[visualization]" --user

Collecting qiskit[visualization]
  Downloading qiskit-0.29.1.tar.gz (12 kB)
  Collecting qiskit-terra==0.18.2
    Downloading qiskit-terra-0.18.2-cp38-cp38-win_amd64.whl (5.3 MB)
  Requirement already satisfied: qiskit-aer==0.8.2 in c:\users\visha\appdata\roaming\python\python38\site-packages (from qiskit[visualization]) (0.8.2)
  Collecting qiskit-ibmq-provider==0.16.0
    Using cached qiskit_ibmq_provider-0.16.0-py3-none-any.whl (235 kB)
  Requirement already satisfied: qiskit-ignis==0.6.0 in c:\users\visha\appdata\roaming\python\python38\site-packages (from qiskit[visualization]) (0.6.0)
  Collecting qiskit-aqua==0.9.5

WARNING: The script f2py.exe is installed in 'C:\Users\visha\AppData\Roaming\Python\Python38\Scripts' which is not on PATH.
```

### 4) If there are no errors, Restart your kernel to finalise installation



## Testing Qiskit Installation

1) Run the cell below to create a quantum circuit with a qubit in superposition

Execute an example quantum program

1) Create a quantum circuit

```
In [1]: # import the objects from qiskit
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit, execute, Aer
from random import randrange

# create a quantum circuit and its register objects
qreg = QuantumRegister(2) # quantum register with two quantum bits
creg = ClassicalRegister(2) # classical register with two classical bit
circuit = QuantumCircuit(qreg,creg) # quantum circuit composed by a quantum register and a classical register

# apply a Hadamard gate to the first qubit
circuit.h(qreg[0])

# set the second qubit to state |1>
circuit.x(qreg[1])

# apply CNOT(first_qubit,second_qubit)
circuit.cx(qreg[0],qreg[1])

# measure the both qubits
circuit.measure(qreg,creg)

print("The execution of the cell was completed, and the circuit was created :)")
```

The execution of the cell was completed, and the circuit was created :)

2) Run the next cell to display the circuit

Run the cell and observe the generated circuit

```
In [2]: # draw circuit
circuit.draw(output='mpl')

# the output will be a "matplotlib.figure" object
```

Out[2]:



The diagram shows a quantum circuit with two qubits, q0 and q1. q0 starts with a Hadamard gate (H) and then a CNOT gate controlled by q0 and targeting q1. q1 starts with an X gate and then a measurement gate. Both qubits end with measurement gates. The classical outputs are c0 and c1.

If there are no errors, your installation is complete and successful! You can run the next code block to execute the circuit and get the results out!

Congratulations! You can continue onto the further notebooks and start your Bronze journey! :D

Prepared by - Vishal Sharathchandra Bajpe



We are working to build an open quantum ecosystem  
More information: [qworld.net](https://qworld.net)