

# 快速使用指南

百度量子计算研究所

2020年9月9日

## 1 总览

### 量易伏概览

量易伏 (英文名: Quantum Leaf) 是百度研究院旗下量子计算研究所开发的云原生 (Cloud Native) 量子计算平台, 提供以量子设施即服务 (Quantum infrastructure as a Service 简称为 QaaS) 模式的量子计算环境。用户可以在量易伏上编程, 并选择在量子模拟器或量子计算机上运行量子程序。为了更好地协助用户进行量子计算实验和算法研究, 量易伏对应不同的用户需求分为以下三个部分:

#### 1. 量子在线平台: PyOnline

PyOnline适用于非工程研究和探索, 有零安装负担的优势。用户只需登录 Quantum-hub 即可享有流畅丰富的量子体验。PyOnline 采用容器化的attach运行方式, 支持自定义文件和自定义Python组件, 使用户能够准确灵活地运用量子计算算力。

#### 2. 量子作曲家: QComposer

QComposer面向初学者和进阶者。用户无需编写代码, 只要拖动相应的组件即可完成量子电路的搭建和验证。Web端将根据搭建的量子电路展示相应代码, 方便用户学习量子编程。

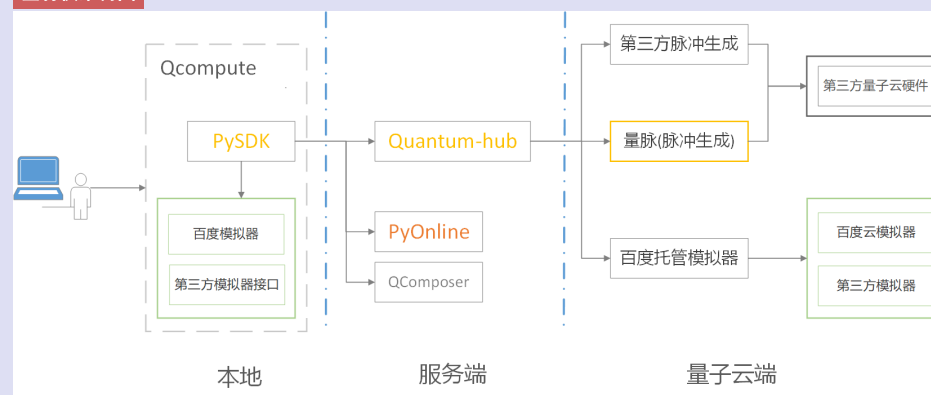
#### 3. 量子开发套件: QCompute

QCompute面向计算机工程师和科学家为主的用户, 提供了全栈式端到端的量子服务。用户端可以混合使用Python编程语言和量子编程语言进行实验和开发。量子端由本地模拟器、在线模拟器和量子计算机组成。本地模拟器分为百度模拟器和第三方模拟器接口, 用户可以在本地直接使用。用户需传入 Quantum-hub 账号对应的Token 远程使用在线模拟器或量子计算机。

“ Quantum mechanics: Real Black Magic Calculus (Albert Einstein) ”

## 用户端-量子端架构

量易伏架构图



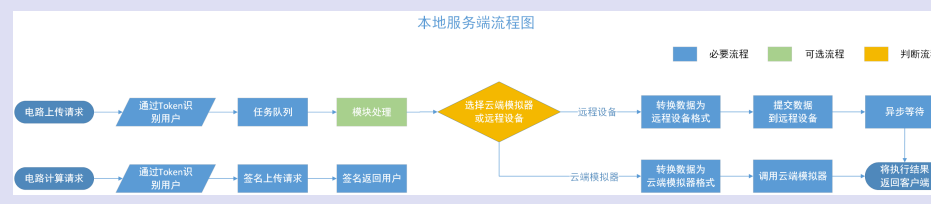
### 1. 用户端可以选用三类不同组件实现编程

- (a) 方便的 PyOnline, 装配SDK组件的在线Python编辑器, 免除配置烦恼, 登录即用。同时, 其高效可扩展架构致力于保护用户数据和知识资产, 保证用户能够上传自定义文件, 下载产生的文件, 定义需要使用的安装包。
- (b) 简单的 QComposer, 通过拖拽量子门图标或者利用QASM交互编程, 便利地完成量子电路搭建。
- (c) 灵活的 QCompute, 拥有最全面的量子计算功能, 携带百度自主研发的轻量级量子模拟器 (Baidu Sim2)、第三方模拟器和连接远程量子云硬件的接口。借助Python灵活的语法和丰富的功能, 实现管理和复用用户的知识资产; 用户可以将代码 (电路) 保存在本地, 也可以提交到云端。

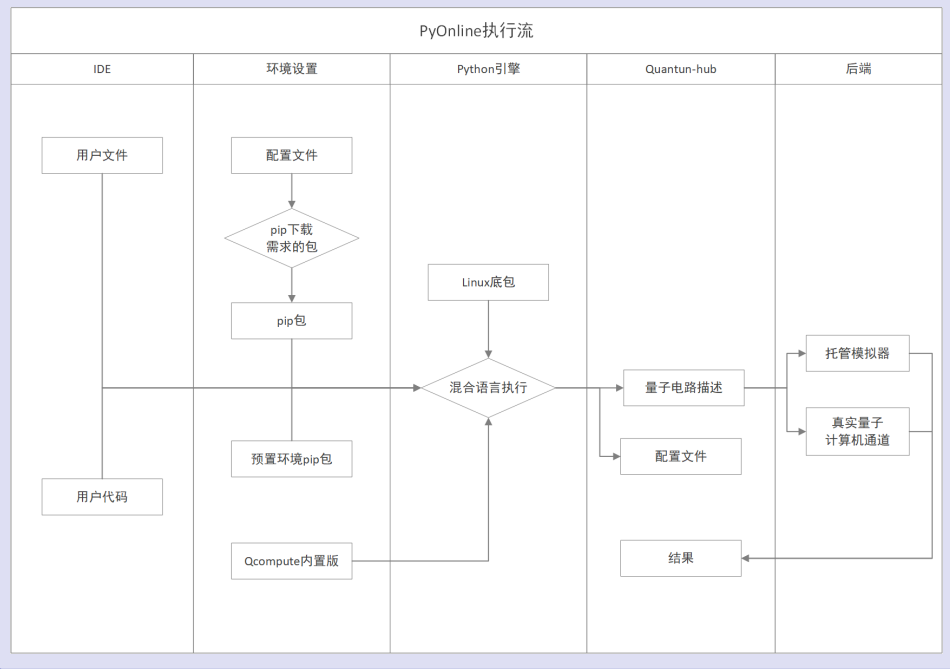
### 2. 量子端有多种量子服务形式可以选择

- (a) **LocalBaiduSim2** 本地运行的百度量子计算研究所使用Python编程语言开发的高效态向量 (State Vector) 模拟器Sim2。
- (b) **CloudBaiduSim2** 云上容器 (Quantum-hub) 内运行的百度量子计算研究所使用Python编程语言开发的高效态向量模拟器Sim2, 百度云计算的算力支持。
- (c) **CloudAerAtBD** 云上容器 (Quantum-hub) 内运行的开源的Aer模拟器 (CPP版本), 百度云计算的算力支持。
- (d) **VIP用户**有优先享有QPU测试使用的权利。

## Client工作流



## PyOnline流程图



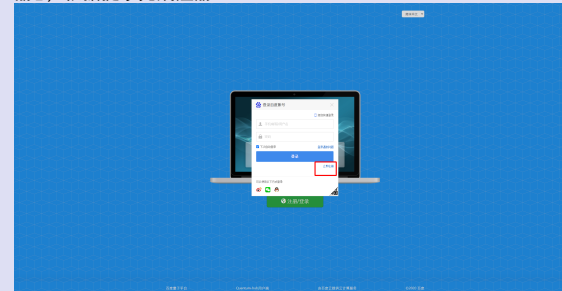
## 2 快速入门

### 新用户引导

量易伏为用户提供了三种方式进行量子计算实验，其中 QCompute 支持本地独立运行和连接在线资源运行，PyOnline 和 QComposer 则需要登录 Quantum-hub 在线运行。在接下来的入门内容中将先引导用户注册 Quantum-hub 和生成 Token（登录后已经生成默认 Token），然后介绍 PyOnline 和 QComposer 的使用，最后介绍 QCompute 的安装和使用。

#### 1. 登录与注册

使用浏览器访问 Quantum-hub，点击页面下方【注册/登录】即可打开登录界面。新用户请点击【立即注册】，根据提示完成注册。

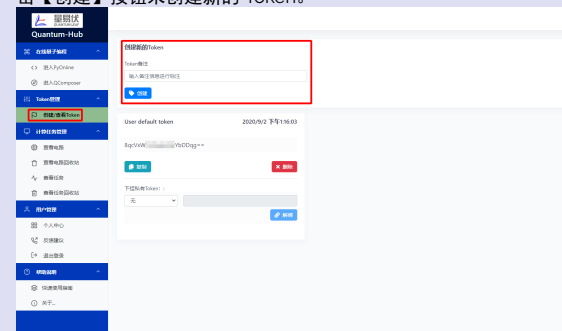


注册完成后，请回到 Quantum-hub 首页，输入用户名和密码后点击【登录】进入量易伏在线平台。

#### 2. 生成Token

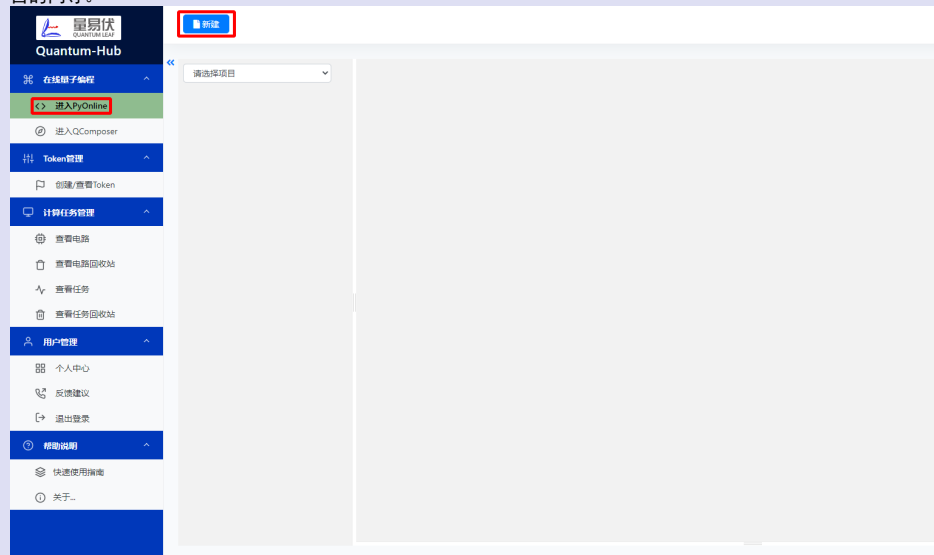
使用系统的第一步是生成 Token。Token 类似于身份证号或账号，是量易伏用于识别用户的标志。当用户在本地图编写代码，但想提交到量易伏的云端服务器执行运算时，用户就需要在本地的代码里输入自己的 Token，Token 请从 Quantum-hub 获取。

登录后，在左侧菜单栏中选择【创建/查看Token】，右侧便会切换到 Token 管理界面。每个用户拥有一个默认的 Token，由大小写字母、数字、等于符号组成。用户也可以在此页面输入【Token备注】并点击【创建】按钮来创建新的 Token。

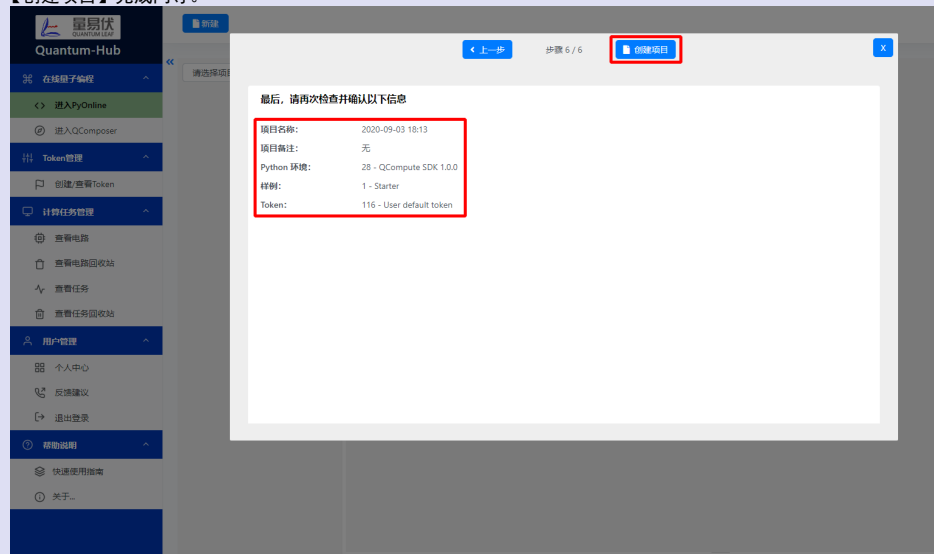


## 快速入门之PyOnline 01

登录后，在左侧菜单栏中点击【进入PyOnline】切换到在线开发界面。点击上方的【新建】按钮后即可打开创建项目的向导。

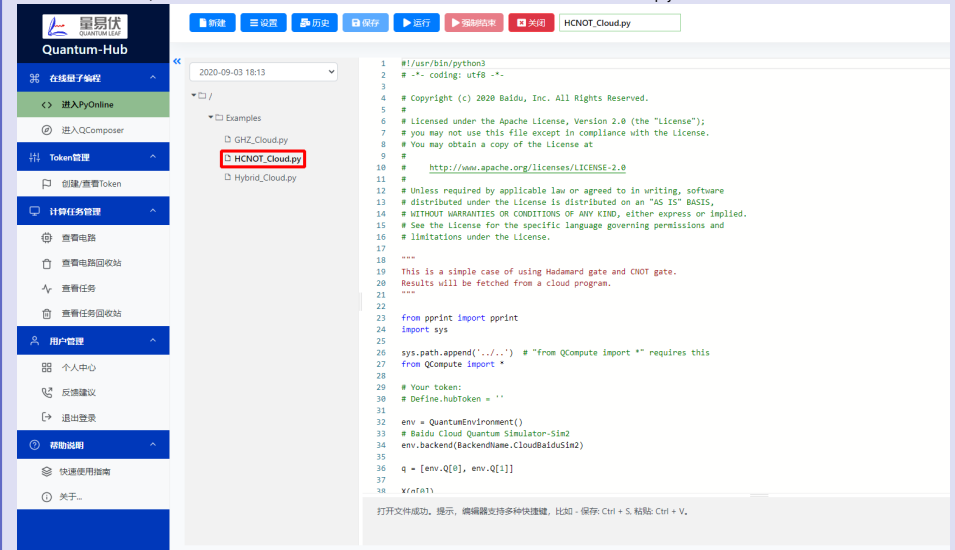


向导由6个步骤组成，用户需按照系统提示依次完成。在步骤6中，请核对之前填写、选择的信息，确认无误后点击【创建项目】完成向导。



## 快速入门之PyOnline 02

项目创建完毕后，可以看到开发界面的左侧有三个实例。双击 HCNOT\_Cloud.py 以查看其中的代码。

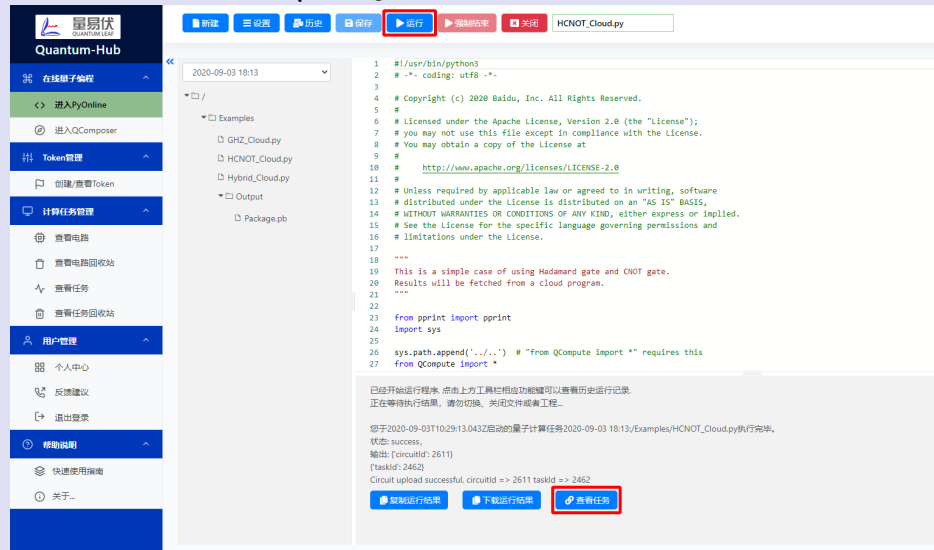


下面的注释解释了 HCNOT\_Cloud.py 中核心代码的作用。

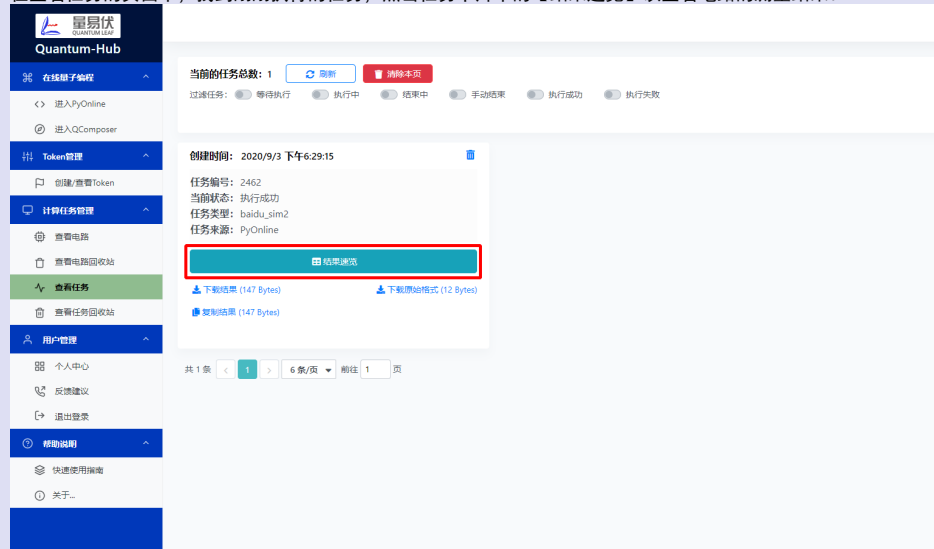
```
1 from QCompute import *
2
3 # 新建一个量子环境
4 env = QuantumEnvironment()
5 # 使用百度自研的量子后端进行模拟
6 env.backend(BackendName.CloudBaiduSim2)
7
8 # 初始化两个量子比特，使其处于基态|00>
9 q = [env.Q[0], env.Q[1]]
10 # 在0号（即第一个）量子比特上加入 X 门
11 X(q[0])
12 # 加入0号控制1号（即第二个）量子比特的 CNOT 门
13 CX(q[0], q[1])
14
15 # 测量第0、1号量子比特
16 MeasureZ(q, range(2))
17 # 测量该电路1024次，结果不返回，请在【查看电路】和【查看任务】中查看
18 taskResult = env.commit(1024, downloadResult=False)
```

## 快速入门之PyOnline 03

点击界面上方的【运行】按钮，稍候片刻，代码下方会出现三个按钮。点击【查看任务】跳转至查看任务的页面。**注意**，每次使用在线资源计算时，都会消耗用户账户中的1 Credit，并且PyOnline本身的运行将额外扣除一个Credit。Credit 使用完后请联系 [quantum@baidu.com](mailto:quantum@baidu.com) 或使用【反馈建议】进行反馈。

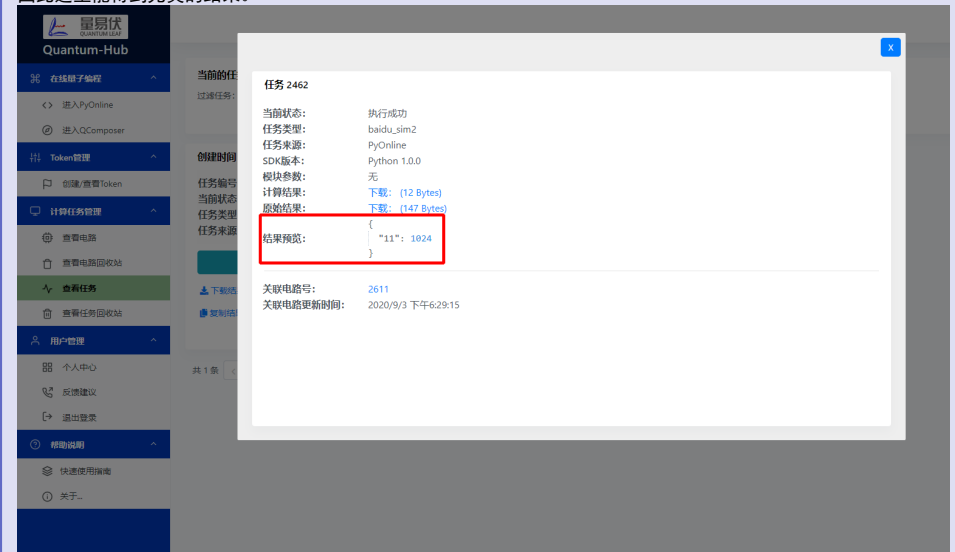


在查看任务的页面中，找到刚刚执行的任务，点击任务卡片中的【结果速览】以查看电路的测量结果。

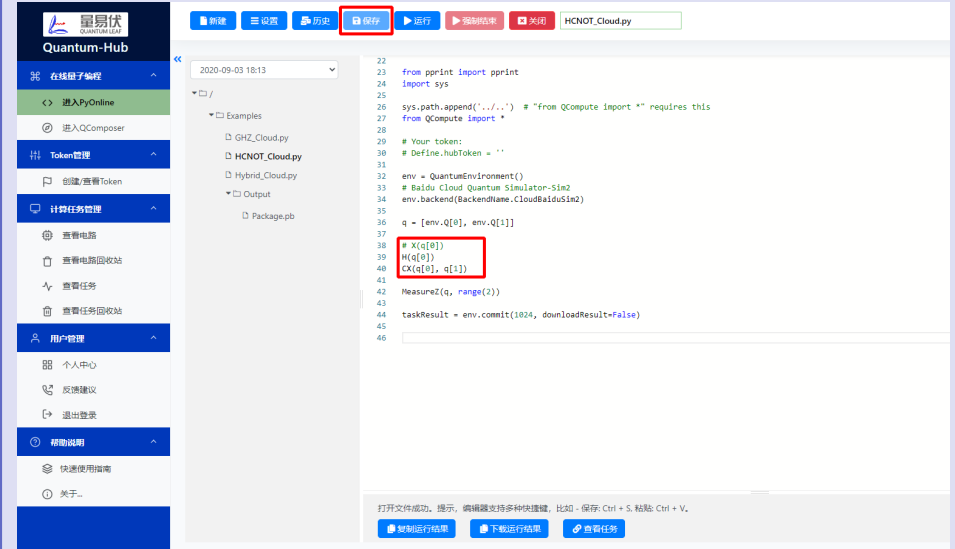


## 快速入门之PyOnline 04

如图所示，对电路进行1024次测量得到的结果均为11，即量子态  $|11\rangle$ 。**注意**，目前百度量子电路模拟不包含噪音，因此这里能得到完美的结果。



接下来试着运行用来生成贝尔态的电路。回到开发环境的界面，将 HCNOT\_Cloud.py 中的 X 门换成 Hadamard 门并保存修改。



## 快速入门之PyOnline 05

运行并查看结果后可以看出，测量得到量子态  $|00\rangle$  和  $|11\rangle$  的次数约各占一半。



任务 2463

当前状态:	执行成功
任务类型:	baidu_sim2
任务来源:	PyOnline
SDK版本:	Python 1.0.0
模块参数:	无
计算结果:	下载: (22 Bytes)
原始结果:	下载: (159 Bytes)
结果预览:	<pre>{   "11": 528,   "00": 504 }</pre>

关联电路号: 2612  
关联电路更新时间: 2020/9/3 下午6:37:58

下面的例子展示了测量函数 MeasureZ0 的用法，用户可以用它测量个别量子比特，例如下面的代码初始化了6个量子比特，但只对第2、4、5号这3个量子比特进行了测量。

```
1 from QCompute import *
2
3 # 新建一个量子环境
4 env = QuantumEnvironment()
5 # 使用百度自研的量子后端进行模拟
6 env.backend(BackendName.CloudBaiduSim2)
7
8 # 初始化6个量子比特，使其处于基态|000000>
9 q = [env.Q[i] for i in range(6)]
10 # 在2号（即第3个）量子比特上加入 Hadamard 门
11 H(q[2])
12 # 加入2号控制4号（即第5个）量子比特的 CNOT 门
13 CX(q[2], q[4])
14
15 # 测量第2、4、5号量子比特：range(3) 中的3代表测量的量子比特的个数
16 MeasureZ([q[2], q[4], q[5]], range(3))
17 # 测量该电路2048次，结果不返回，请在【查看电路】和【查看任务】中查看
18 taskResult = env.commit(2048, downloadResult=False)
```

## 快速入门之PyOnline 06

理论上，测量得到的量子态中约一半是  $|000\rangle$ ，另一半是  $|110\rangle$ ，但实际的运行结果显示，000和011各占了约一半的次数。



任务 2464

当前状态:	执行成功
任务类型:	baidu_sim2
任务来源:	PyOnline
SDK版本:	Python 1.0.0
模块参数:	无
计算结果:	下载: (26 Bytes)
原始结果:	下载: (163 Bytes)
结果预览:	<pre>{   "000": 1030,   "011": 1019 }</pre>

关联电路号: 2613  
关联电路更新时间: 2020/9/3 下午6:39:28

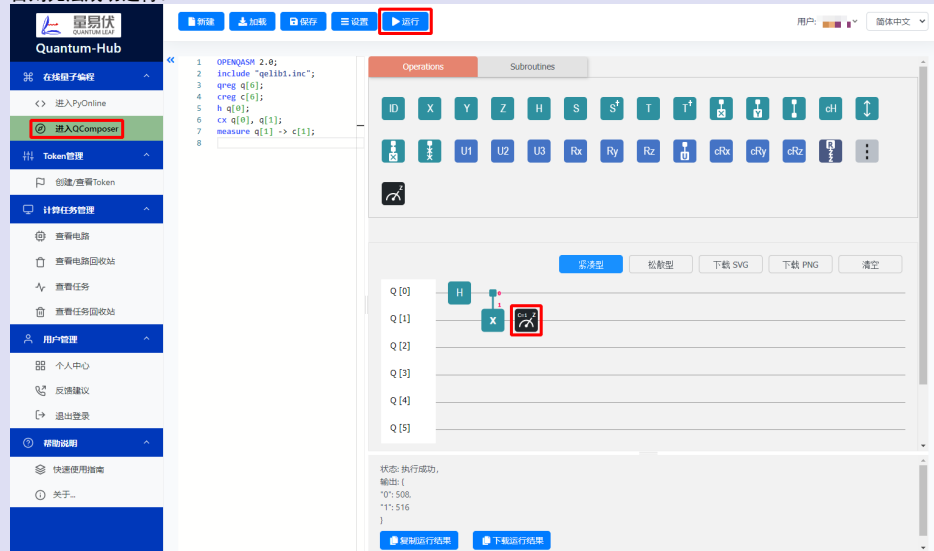
其实，结果预览中的011和量子态  $|110\rangle$  是等价的。常用的教科书采用小端在前的标记方法，即  $|q_0 q_1 q_2 \dots\rangle$ ，但是量易伏在输出测量结果时采用大端在前的标记方法，即  $q_n q_{n-1} q_{n-2} \dots$ 。用户可以运行以下代码以便于更好地理解量易伏采用的标记方法。

```
1 from QCompute import *
2
3 # 新建一个量子环境
4 env = QuantumEnvironment()
5 # 使用百度自研的云量子后端进行模拟
6 env.backend(BackendName.CloudBaiduSim2)
7
8 # 初始化6个量子比特，使其处于基态|000000>
9 q = [env.Q[index] for index in range(6)]
10 # 在0号（即第1个）量子比特上加入 X 门
11 X(q[0])
12
13 # 测量所有量子比特
14 MeasureZ(q, range(6))
15 # 测量该电路2048次，结果不返回，请在【查看电路】和【查看任务】中查看
16 taskResult = env.commit(2048, downloadResult=False)
```

运行上述代码得到的2048次测量的结果均为000001，对应量子态  $|100000\rangle$ 。

## 快速入门之QComposer

点击左侧菜单栏中的【进入QComposer】即可进入 QComposer 界面，用户可以使用拖拽的方法在量子电路上添加量子门。搭建好电路后，点击界面上方的【运行】即可对该电路进行测量。注意，电路最后必须加上测量模块，否则无法成功运行。



以上就是 PyOnline 和 QComposer 的快速入门，用户还可以运行一些更有意思的例子，例如 VQE (Variational Quantum Eigensolver) 和 Grover 算法等。

## 快速入门之QCompute O1

QCompute是一个可以在本地独立使用的量易伏量子计算二次开发工具，也可通过其在本地调用百度云端的计算资源。

1. 安装Python环境  
安装 Python 3.6 及以上版本的运行环境。

2. 使用PIP安装QCompute  
执行

```
pip install qcompute
```

或

```
pip install -e .
```

3. 验证安装的正确性  
安装完成后，运行下列代码以确保 QCompute 已正确安装：

```
python -m QCompute.Test.PostInstall.PostInstall
```

稍候片刻，如输出：

```
Local test succeeded.
```

则表示本地模拟器没有问题。随后提示：

```
Please provide a token:
```

时，请输入从 Quantum-hub 生成的 Token。稍候片刻，如输出：

```
Cloud test succeeded.
```

则表示云端模拟器没有问题。

## 快速入门之QCompute 02

新建bell.py文件后，导入需要使用的包

```
from pprint import pprint
import sys
sys.path.append('../..') # from QCompute import *; Needing
from QCompute import *
```

定义Token，注：Token 唯一标识了电路提交者。用户使用 QCompute 连接远程模拟器或设备进行实验时，一定不要忘记填入，Quantum-hub 上的 PyOnline 已经完全内置写入不需要用户自行填入。GPU 仅限 VIP 用户使用。

```
Define.hubToken = ''
```

定义量子计算环境

```
env = QuantumEnvironment()
# Baidu Cloud Quantum Simulator-Sim2
env.backend(BackendName.CloudBaiduSim2)
```

设置量子比特

```
q = [env.Q[0], env.Q[1]]
```

生成量子电路

```
H(q[0])
CX(q[0], q[1])
MeasureZ(q, range(2))
```

提交量子电路，获取并输出结果

```
taskResult = env.commit(1024, fetchMeasure=True)
pprint(taskResult)
```

至此贝尔态的制备已完成。

## Quantum-hub与QCompute的区别

Quantum-hub 是量易伏中控端，无论云端 PyOnline、QComposer 或是本地运行的 QCompute 结果都可以从 Quantum-hub 获取。QCompute 是量易伏的一个高级开发组件，通过此组件，用户可以进行Python与简单的量子编程语言的混合编程等。在云端部署的 PyOnline 也内含 QCompute 组件。

下面展示一个贝尔态电路测量的例子。

首先，调用云端的服务器运行下列代码，每次执行都会消耗用户账户中的 1 Credit，并且PyOnline本身的运行额外扣除一个Credit。注意，使用在线资源时，用户必须先登录 Quantum-hub 生成并拷贝 Token，粘贴到运行代码的

Define.hubToken = '' 一行中。

```
1 from pprint import pprint
2 import sys
3 sys.path.append('../..')
4 from QCompute import *
5
6 # 新建一个量子环境
7 Define.hubToken = '用户的Token'
8 # 使用百度的量子云端进行模拟
9 env = QuantumEnvironment()
10 env.backend(BackendName.CloudBaiduSim2)
11
12 # 初始化两个量子比特，使其处于基态|00>
13 q = [env.Q[0], env.Q[1]]
14 # 在0号（即第一个）量子比特上加入 Hadamard 门
15 H(q[0])
16 # 加入0号控制1号（即第二个）量子比特的 CNOT 门
17 CX(q[0], q[1])
18
19 # 测量第0、1号量子比特
20 MeasureZ(q, range(2))
21 # 测量该电路2048次并将结果保存在 taskResult 中
22 taskResult = env.commit(2048, fetchMeasure=True)
23 pprint(taskResult)
```

现在，改用 QCompute 里面的本地端模拟器执行同样的任务，此时不会消耗用户账户中的Credit。

```
1 from pprint import pprint
2 import sys
3 sys.path.append('../..')
4 from QCompute import *
5
6 # 新建一个量子环境
7 env = QuantumEnvironment()
8 # 使用百度的量子本地端进行模拟
9 env.backend(BackendName.LocalBaiduSim2)
10
11 # 初始化两个量子比特，使其处于基态|00>
12 q = [env.Q[0], env.Q[1]]
13 # 在0号（即第一个）量子比特上加入 Hadamard 门
14 H(q[0])
15 # 加入0号控制1号（即第二个）量子比特的 CNOT 门
16 CX(q[0], q[1])
17
18 # 测量第0、1号量子比特
19 MeasureZ(q, range(2))
20 # 测量该电路2048次并将结果保存在 taskResult 中
21 taskResult = env.commit(2048, fetchMeasure=True)
22 pprint(taskResult)
```

相较于本地执行，调用云端服务器时尽管返回结果的速度稍慢，但其计算力远大于个人电脑。当用户需要模拟几十个量子比特时，建议使用量易伏的云端服务器。当然，如果用户只需要执行一些简单且规模小的任务，使用个人电脑完全没有问题。

以上就是量易伏的快速入门。在文件夹中还有一些常见量子算法的教程，用户可以自行学习探索。

### 3 工具集

#### 快速调整参数

是否打印每次**fetch**结果 ..... Define/Settings/outputInfo  
输出进制为**16**进制或者**2**进制 ..... Define/Settings/measureFormat  
**Sim2**模拟器进程内启动（速度）或外部启动（稳定） ..... OpenSimulator/local\_baidu\_sim2/commit  
**backend**可用列表 ..... QuantumPlatform/\_\_\_init\_\_\_/BackendName

#### FAQ

1. 如何在量易伏上编程？
  - (a) 用户可以混合使用Python编程语言和简单的量子编程语言进行实验和开发；
  - (b) 如用户使用 Quantum-hub 上的 QComposer 或 PyOnline 进行实验，平台自动调用 Token，不需要用户自行填入；
  - (c) 如用户使用 QCompute 进行实验，需要使用上述示例代码传入 Token 以连接远端模拟器或设备。
2. 为什么要用 QCompute ？
  - (a) 可以迁移使用经典语言知识；
  - (b) 能够实现例如变分算法这样需要根据当前的结果调整后续计算的算法。
3. 本地模拟器和在线模拟器有什么区别？
  - (a) 本地模拟器使用本地资源，在线模拟器使用百度云计算资源；
  - (b) 在线模拟器在百度云计算的支持下提供更为强大的算力。
4. 如何加速计算过程？
  - (a) 使用在线模拟器；
  - (b) 如果是交互式计算，考虑使用并行计算方法。
5. Credit的用处？
  - (a) 在线资源计算的时候每次会消耗1 Credit，PyOnline本身的运行将额外扣除 1 Credit；
  - (b) 使用完后请联系 [quantum@baidu.com](mailto:quantum@baidu.com) 或者 Quantum-hub 中的“反馈建议”联系我们获取更多点数。
6. 如何注册VIP？

VIP注册为邀请制，用户需要完成：

  - (a) 相应身份核验；
  - (b) 提交VIP申请到邮箱 [quantum@baidu.com](mailto:quantum@baidu.com) 或 Quantum-hub 中的“反馈建议”；
  - (c) 待后台判断通过，则申请成功。