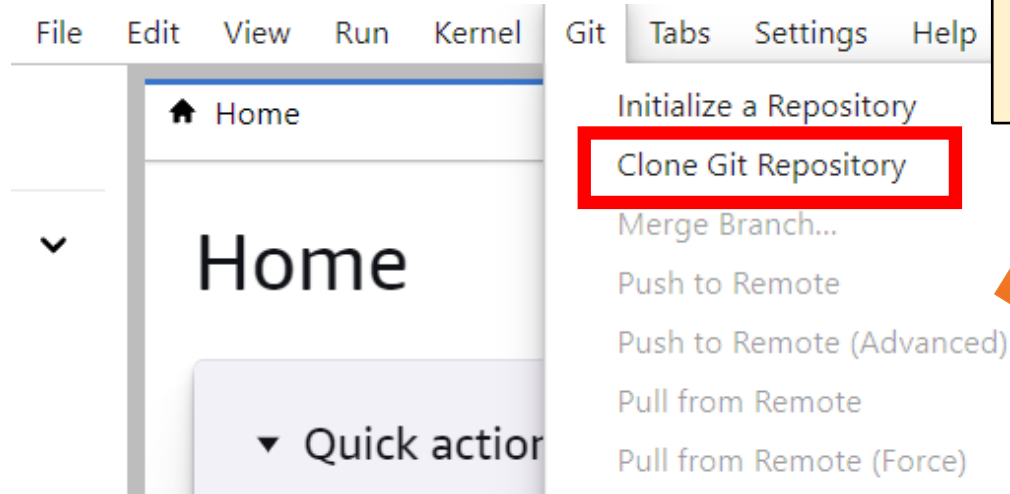


# ハンズオン環境の起動①

今回利用するTKETの教材は、  
これまでの量子ソフトウェア勉強会の教材とは  
別のGitHubから取得します。

GitHubからハンズオン教材を、ユーザ毎の環境に取得



メニューから  
「Git > Clone Git Repository」をクリック

Clone Git Repository

Git repository URL (.git):

<https://github.com/cqcjapan/QSRH-Quantinuum.git>

Project directory to clone into:

/path/to/local/directory or empty for the root directory of JupyterLab

次のURLを入力すると現れる「Clone～」の文字をクリック  
<https://github.com/cqcjapan/QSRH-Quantinuum.git>

Cancel

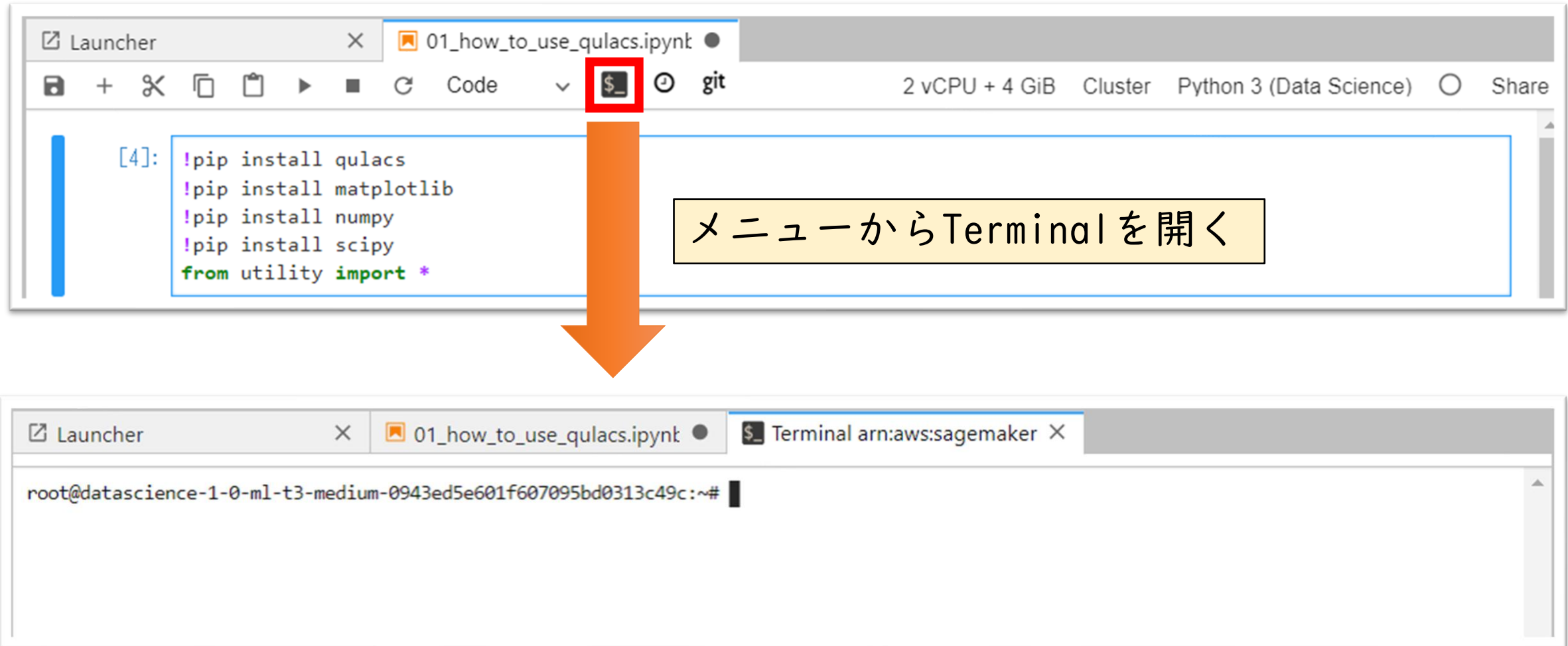
Clone

「Clone」をクリック

## ハンズオン環境の起動②

今回利用するTKETの教材は、  
これまでの量子ソフトウェア勉強会の教材とは  
別のGitHubから取得します。

GitHubからハンズオン教材を、ユーザ毎の環境に取得



The screenshot illustrates the steps to open a terminal in the JupyterLab environment. The top panel shows the 'Code' view with a red box around the terminal icon in the toolbar. An orange arrow points down to the bottom panel, which shows the 'Terminal' view with a shell prompt.

Top Panel (Code View):

- Toolbar: `$` icon (highlighted with a red box)
- Code Editor: 

```
[4]: !pip install qulacs
!pip install matplotlib
!pip install numpy
!pip install scipy
from utility import *
```

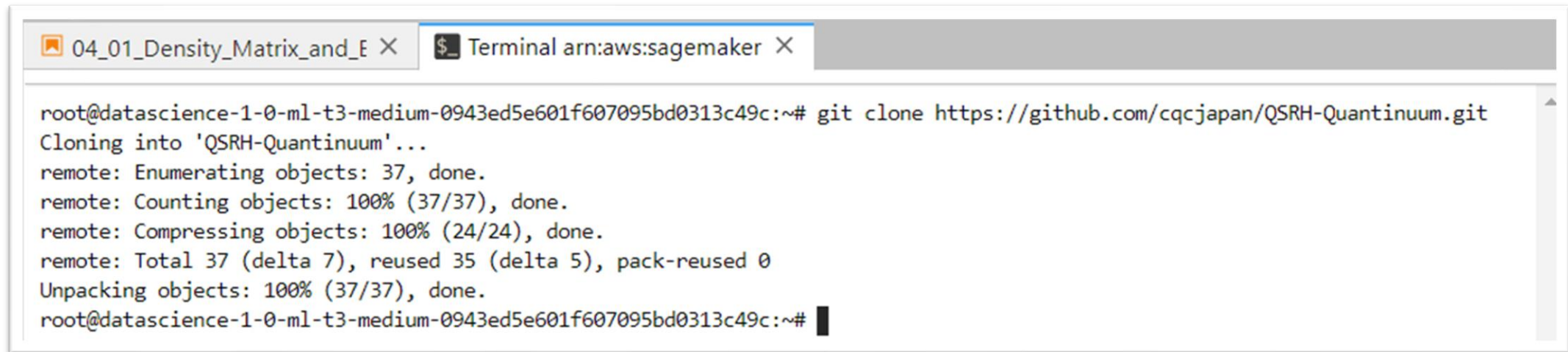
Bottom Panel (Terminal View):

- Terminal Title: `$ Terminal arn:aws:sagemaker`
- Terminal Prompt: `root@datascience-1-0-ml-t3-medium-0943ed5e601f607095bd0313c49c:~#`

メニューからTerminalを開く

## ハンズオン環境の起動②

gitコマンドを使って、TKETのハンズオン教材を取得



```
04_01_Density_Matrix_and_E X Terminal arn:aws:sagemaker X
root@datascience-1-0-ml-t3-medium-0943ed5e601f607095bd0313c49c:~# git clone https://github.com/cqcjapan/QSRH-Quantinuum.git
Cloning into 'QSRH-Quantinuum'...
remote: Enumerating objects: 37, done.
remote: Counting objects: 100% (37/37), done.
remote: Compressing objects: 100% (24/24), done.
remote: Total 37 (delta 7), reused 35 (delta 5), pack-reused 0
Unpacking objects: 100% (37/37), done.
root@datascience-1-0-ml-t3-medium-0943ed5e601f607095bd0313c49c:~#
```

```
# git clone https://github.com/cqcjapan/QSRH-Quantinuum.git
```

Teamsのファイルビューワーからこのファイルを開いてURLをコピーすると、末尾に余分な空白が入り、エラーとなるケースがあります。

URLを一度テキストエディタなどにコピーし、正しいURLを入力すると安全です。

# ハンズオン環境の起動

## 教材が置いてある場所に移動

The image shows a file browser window on the left and a presentation slide on the right.

**File Browser:**

- Search bar: Filter files by name
- Path: / QSRH-Quantinuum / 2023-09-01QSRH /
- Table with columns: Name, Last Modified
- Files listed:
  - fig (19 hours ago)
  - key (19 hours ago)
  - 20220901-QSRH2.ipynb (19 hours ago) - **Highlighted with a red box**
  - 20230901-QSRH1.ipynb (19 hours ago)

**Presentation Slide:**

- Tab: tket\_cheat\_sheet-ver1.3(tket' X
- Header: PowerPoint プレゼンテーション | 1 / 2 | 60%
- Title: Quantum SDK TKET :: CHEAT SHEET
- Diagram: Pipeline of quantum computing
  - ① Install pytket and pytket-extensions
  - ② Create quantum circuit
  - ③ Compile quantum circuit (simplify)
  - ④ Compile quantum circuit (fit device architecture)
  - ⑤ Execute quantum circuit on quantum device
- Content sections:
  - Install pytket and pytket-extensions**
    - An implementation of TKET is currently available in the form of the `pytket` package for python 3.9+ on Linux, MacOS and Windows.
    - `pip install pytket`
    - `pytket-extension` modules are available for interfacing pytket with several quantum software, including Qiskit, Cirq, and for adding quantum devices and simulators to target.
    - Each extension module can be installed similarly as follows.
    - `pip install pytket-X`
    - Ex. when you install TKET extension for qiskit
    - `pip install pytket-qiskit`
    - A full list of available pytket extensions is shown in the webpage
  - Supported quantum\_gates:**
    - `{X, Y, Z, H, S, Sdg, T, Tdg, SX, SXdg, V, Vdg, SWAP, ISWAPMax, ECR}`
    - Ex. Apply X gate to the qubit q[0] and SWAP gate to qubits (q[1],q[2]).
    - `circ.X(0)`
    - `circ.SWAP(1,2)`
  - Supported controlled quantum\_gates:**
    - `{CX, CY, CZ, CH, CSX, CSXdg, CV, CVdg, CSWAP, CCX}`
    - Ex. Apply CX gate to the circuit with control qubit q[0] and target qubit q[1], CCX gate with control qubits (q[0],q[2]) and target qubit q[1], and CSWAP gate with control qubit q[2] and target qubits (q[0],q[1]).
    - `circ.CX(0,1)`
    - `circ.CCX(0,2,1)`
  - Measurement**
    - `Circuit.Measure` appends a single qubit Z-basis measurement.
    - `Circuit.measure_register` appends a measure gate to all qubits in the given register, storing the results in the given classical bits.
    - `Circuit.measure_all` appends a measure gate to all qubits, storing the results in classical bits.
    - Ex. Append measure gate to q[1], storing result in c[0], append measure gates to all qubits in the qubit register with name 'q', storing the bit register with name 'c', and append measure gates to all qubits in the circuit.
    - `circ.Measure(1, 0) #measure gate to q[1], storing c[0]`
    - `circ.measure_register(circ.get_q_register('q'), 'c') #measure q reg`
    - `circ.measure_all() #measure all qubits in the circuit`
  - Quantum Circuit Visualisation**

ファイルブラウザで移動

QSRH-Quantinuum > 2023-09-01QSRH

# ハンズオン環境の起動

## 開く教材をダブルクリック

Filter files by name

/ QSRH-Quantinuum / 2023-09-01QSRH /

Name	Last Modified
fig	19 hours ago
key	19 hours ago
20220901-QSRH2.ipynb	19 hours ago
20230901-QSRH1.ipynb	19 hours ago



Set up notebook environment

Set up environment for "01\_00\_quantum\_computation\_1000practices.ipynb".

Image: Base Python 3.0

Kernel: Python 3

Instance type: ml.t3.medium

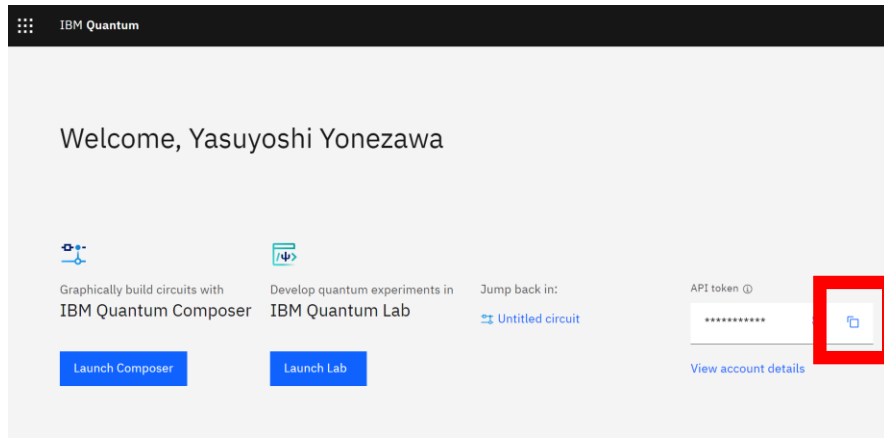
Start-up script: No script

Cancel Select

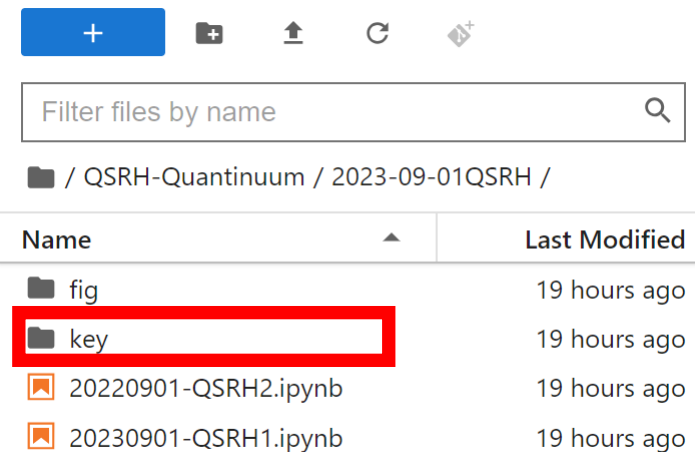
Base Python 3.0 のImageを選択し  
「Select」をクリック  
(Python 3.9+であればなんでも良い)

# IBMQ API tokenの設定

<https://quantum-computing.ibm.com/> にアクセス  
(アカウントがなければ、各自が作成する必要があります。無料です)



クリックしてtokenをコピー



Keyフォルダーのibm-tokenファイルを開きペースト