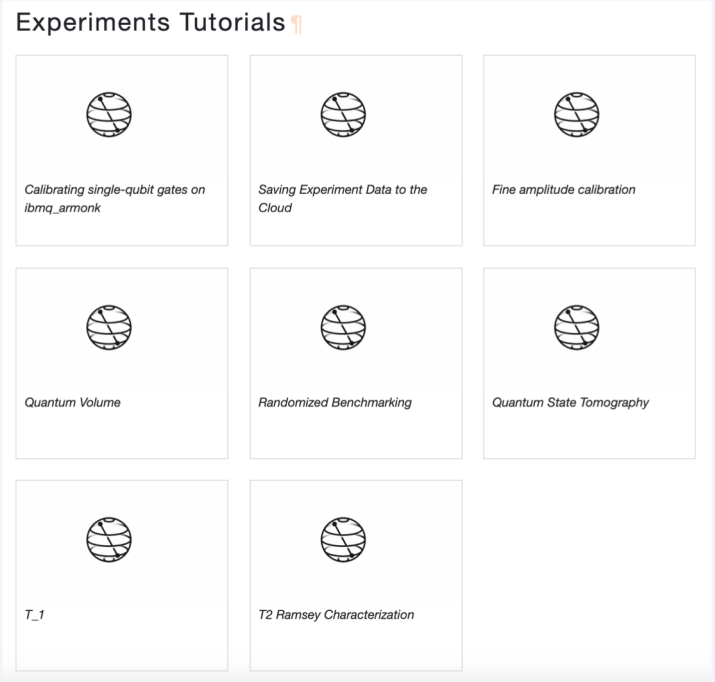
#44 Update qiskit-experiment tutorials

Jale Ipekoglu

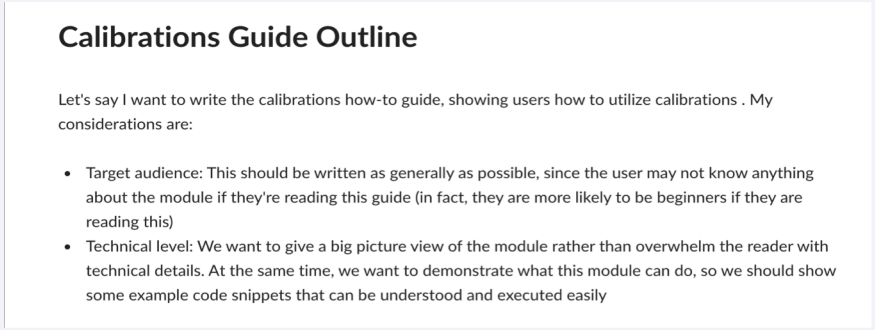
Current Tutorials > Ultimate Goal

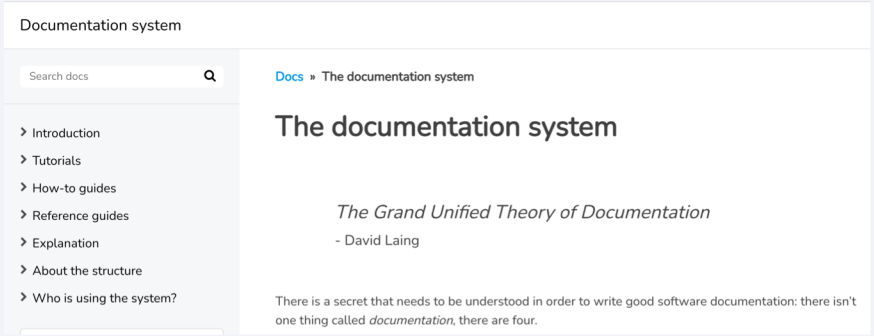
- Step-by-step progression
- Regroup and reformat
- Goal-oriented how-to guides



Steps

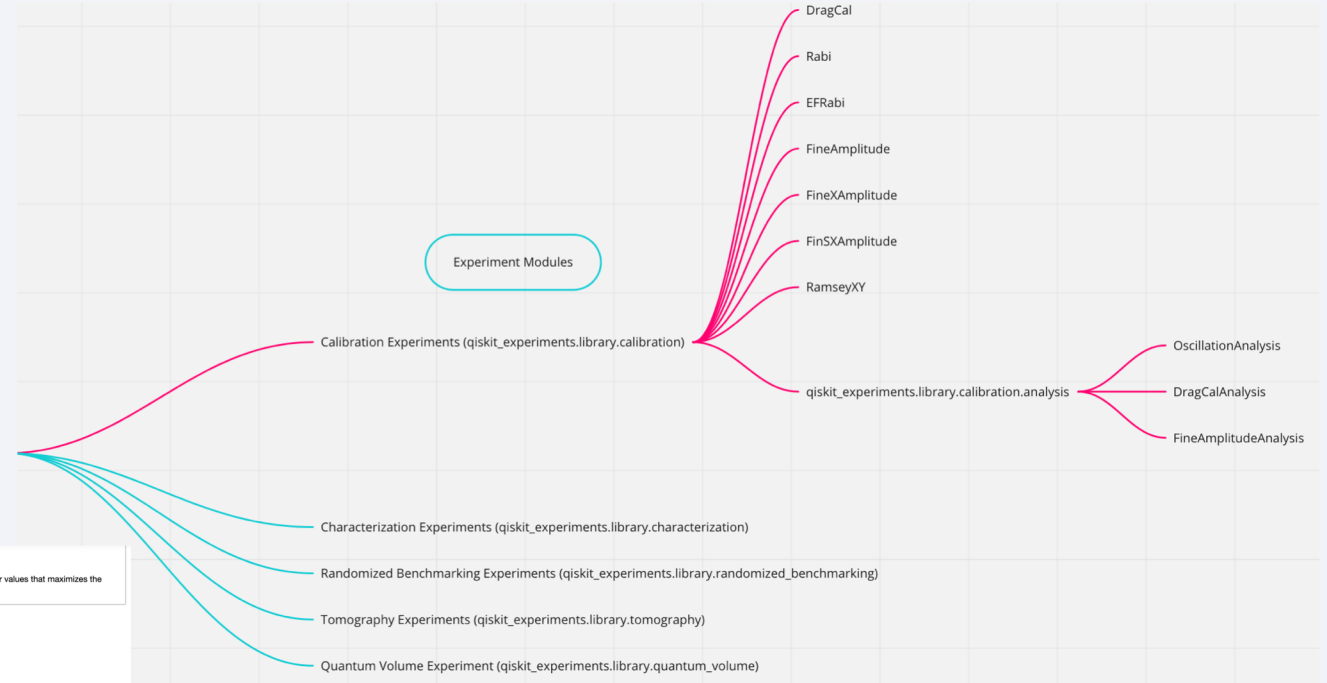
- Create a new guide plan
- Define the outline
- Set the vocabulary

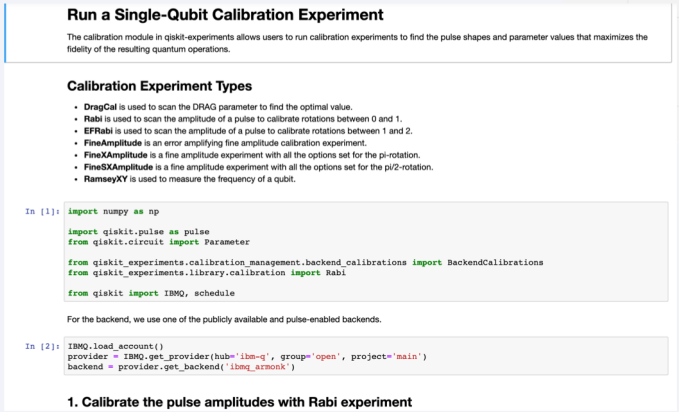




Progress

- Start from API reference
- Decide on prioritization
- Writing the guides





Run a Single-Qubit Calibration Experiment

The calibration module in qiskit-experiments allows users to run calibration experiments to find the pulse shapes and parameter values that maximizes the fidelity of the resulting quantum operations.

Calibration Experiment Types

- **DragCal** is used to scan the DRAG parameter to find the optimal value.
- **Rabi** is used to scan the amplitude of a pulse to calibrate rotations between 0 and 1.
- **EFRabi** is used to scan the amplitude of a pulse to calibrate rotations between 1 and 2.
- **FineAmplitude** is an error amplifying fine amplitude calibration experiment.
- **FineXAmplitude** is a fine amplitude experiment with all the options set for the pi-rotation.
- **FineSXAmplitude** is a fine amplitude experiment with all the options set for the pi/2-rotation.
- **RamseyXY** is used to measure the frequency of a qubit.

In [1]:
```
import numpy as np

import qiskit.pulse as pulse
from qiskit.circuit import Parameter

from qiskit_experiments.calibration_management.backend_calibrations import BackendCalibrations
from qiskit_experiments.library.calibration import Rabi

from qiskit import IBMQ, schedule
```

For the backend, we use one of the publicly available and pulse-enabled backends.

In [2]:
```
IBMQ.load_account()
provider = IBMQ.get_provider(hub='ibm-q', group='open', project='main')
backend = provider.get_backend('ibmq_armonk')
```

1. Calibrate the pulse amplitudes with Rabi experiment

Jale Ipekoglu