opentrons

We thought you'd like to know...

The documentation you are viewing is for the Opentrons OT-1. If you have an Opentrons OT-2, please click here for the OT-2 Python Protocol API documentation.

Opentrons API

The Opentrons API is a simple framework designed to make writing automated biology lab protocols easy.

We’ve designed it in a way we hope is accessible to anyone with basic computer and wetlab skills. As a bench scientist, you should be able to code your automated protocols in a way that reads like a lab notebook.

View source code on GitHub

How it Looks

The design goal of the Opentrons API is to make code readable and easy to understand. For example, below is a short set of instruction to transfer from well `'A1'` to well `'B1'` that even a computer could understand:

```
Use the Opentrons API's containers and instruments

Add a 96 well plate, and place it in slot 'B1'
Add a 200uL tip rack, and place it in slot 'A1'

Add a 200uL pipette to axis 'b', and tell it to use that tip rack

Transfer 100uL from the plate's 'A1' well to it's 'A2' well
```

If we were to rewrite this with the Opentrons API, it would look like the following:

```
# imports
from opentrons import containers, instruments

# containers
plate = containers.load('96-flat', 'B1')
tiprack = containers.load('tiprack-200ul', 'A1')

# pipettes
pipette = instruments.Pipette(axis='b', max_volume=200, tip_racks=[tiprack])

# commands
pipette.transfer(100, plate.wells('A1'), plate.wells('B1'))
```

How it’s Organized

When writing protocols using the Opentrons API, there are generally three sections:

1. Imports
2. Containers
3. Pipettes

When writing in Python, you must always include the Opentrons API within your file. We most commonly use the `containers` and `instruments` sections of the API.

From the example above, the “imports” section looked like:

```
from opentrons import containers, instruments
```

Containers

While the imports section is usually the same across protocols, the containers section is different depending on the tip racks, well plates, troughs, or tubes you’re using on the robot.

Each container is given a type (ex: `'96-flat'`), and the slot on the robot it will be placed (ex: `'B1'`).

From the example above, the “containers” section looked like:

```
plate = containers.load('96-flat', 'B1')
tiprack = containers.load('tiprack-200ul', 'A1')
```

Pipettes

Next, pipettes are created and attached to a specific axis on the OT-One (`'a'` or `'b'`). Axis `'a'` is on the center of the head, while axis `'b'` is on the left.

There are other parameters for pipettes, but the most important are the `max_volume` to set it’s size, and the tip rack(s) it will use during the protocol.

From the example above, the “pipettes” section looked like:

```
pipette = instruments.Pipette(axis='b', max_volume=200, tip_racks=[tiprack])
```

Commands

And finally, the most fun section, the actual protocol commands! The most common commands are `transfer()`, `aspirate()`, `dispense()`, `pick_up_tip()`, `drop_tip()`, and much more.

This section can tend to get long, relative to the complexity of your protocol. However, with a better understanding of Python you can learn to compress and simplify even the most complex-seeming protocols.

From the example above, the “commands” section looked like:

```
pipette.transfer(100, plate.wells('A1'), plate.wells('B1'))
```