How to Opentrons

Parker Ackerknecht

Contents

1 Python Protocol Design

This protocol style uses the python code to run OPT. This code can be simulated as well as designed without the need for an internet connection. When creating a .py file to interface with OPT you will need to import the correct libraries and set metadata for the file.

Note: The current quotation marks in these code snippets tend to be viewed as the wrong style quotation marks for some reason.

1.1 Python Protocol Setup

The use of the metadata section allows for us to control what is seen in the OT-2 application and allows for users to get information for each imported file without the original file.

```
from opentrons import protocol_api
metadata = {
    'protocolName': 'Same Transfer',
    'author': 'Parker',
    'description': 'Transfer from one well to same well on other
    plate',
    'source': 'not needed, but sometimes nice',
    'apiLevel': '2.10' #this is the apilevel that OPT uses
}
```

1.1.1 In-Line

With this method you must type out the proper name of the equipment out each time you refer to the labware, while the Json method makes it possible to have referable variables. Both methods you need to set labware location as that is the only way that OPT will be able to find the labware position. It is also important to use the correct labware, since each " named value stores measurement data for each specific labware (volume, height, etc.)

```
def run(protocol):
    tips=[protocol.load_labware('opentrons_96_tiprack_20ul', '1')]
    #tips are the only load_labware command that need brackets
    around it, no idea why that is but I guess good to know
    pipette = protocol.load_instrument('p20', 'left', tip_racks=
    tips)
```

All OT-2 python protocols include a run function. This is the main source of command for OPT, so all base functions are within this function.

Note: You can call the function something other than protocol. When you have to call commands later with protocol.commands() you have to use the name within the run() example: run(name) is name.commands() throughout the protocol (though still suggest going with protocol and naming the pipette value pip or pipette)

1.1.2 Json Library

In this method you are making a function to import the json values into variable names i.e. what pipette is being used single 20 uL pipette in given code below. *Note:* This code can go before or after the metadata function, but must come before the run protocol.

```
def get_values(*names):
    import json
    _all_values = json.loads("""{"_pip_model":"p20_single_gen2","
    _pip_mount":"right","well_1":"nest_96_wellplate_200ul_flat","
    well_2":"nest_96_wellplate_200ul_flat"}""")
    return [_all_values[n] for n in names]
```

The values within the brackets example [pip_model…] can be set to any variable name that you would like. The get_value('pip_model'…) need to be the same as the variable set in the get_value function.

```
def run(protocol):
    [pip_model, pip_mount, well_1, well_2] = get_values ( '
    _pip_model', '_pip_mount', 'well_1', 'well_2')
tips=[protocol.load_labware('opentrons_96_tiprack_20ul', '1')]
    pipette = protocol.load_instrument(pip_model, pip_mount,
    tip_racks=tips)
```

This code sets up tip location as well as pipette status. You can also define tip_type in get_values, but you usually only have one tip tray so doing the json method is a bit overkill.

1.2 Python Transfer Command

Link to Transfer Command Article

You will need to set up location of each labware used, as well as the pipette commands in order for OPT to know what to do.

```
    plate_1 = protocol.load_labware(well_1,'2').wells()
plate_2 = protocol.load_labware(well_2,'3').wells()
for i, j in zip(plate_1, plate_2)
    pipette.transfer(volume, i,j)
```

This code goes from the first well A1 to the last well H12 and transfers the volume selected in a one to one ratio to plate 1 well A1 to plate 2 well A1. In order to do specific wells:

```
plate_1 = protocol.load_labware(well_1,'2').wells('A1')
plate_2 = protocol.load_labware(well_2,'3').wells('A1')
pipette.transfer(volume, plate_1, plate_2)
```

Instead of .wells() to dictate the wells that you want to use. You can use .rows() or .columns(). The main difference with .wells() is that .wells() uses '' to find location of the wells while .rows() .columns() do not.

```
pip.wells('A1')
pip.rows(2)#rows align with the letters
pip.columns(2) # columns align with the numbers
```

Note: If you really wanted to, you can do negative numbers which is just right to left on the plate instead of left to right. So .rows(-1) would be H.

Two useful commands that are categorized as transfer commands are pip.consolidate(volume, destination, aspirate) and pip.distribute (volume, dest, asp). These two commands work the best with rows() or columns() for quick access since you don't have to set an array for specific wells (you can but takes a lot more work).

```
plate_1 = protocol.load_labware(well_1,'2').rows(2)
    plate_2 = protocol.load_labware(well_2,'3').wells('A1')

    pipette.consolidate(2,plate_1, plate_2)
```

```
plate_1 = protocol.load_labware(well_1,'2').rows(3)
    plate_2 = protocol.load_labware(well_2,'3').wells('A1')

    pipette.distribute(2,plate_2,plate_1)
```

1.2.1 Special Transfer Commands

Link to Advanced Transfer Commands Article

One of the ways to do the mix command is to use the mix_before and mix_after commands in line with the transfer command, with =(times, volume). *Note:* The volume cannot be more than the max volume of the pipette tip.

```
pipette.transfer(100, plate.wells('A1'), plate.wells('A2'),
    mix_before=(2, 50), # mix 2 times with 50uL before aspirating
    mix_after=(3, 75))  # mix 3 times with 75uL after dispensing
```

There is also the blow_out variable that exists to completely empty the pipette tip.

```
pipette.transfer(100,plate.wells('A1'),plate.wells('A2'),
    blow_out=True)      # blow out droplets when tip is empty
```

In the same vein you have the touch_tip command that if set to True the pipette will touch each well's edge that it is transferring to.

```
pipette.transfer(100,plate.wells('A1'),plate.wells('A2'),
    touch_tip=True)     # touch tip to each well's edge
```

There is also the trash command that if set to False will make the tip return to the rack and not be dispensed into the trash bin

```
pipette.transfer(100,plate.wells('A1'),plate.wells('A2'),
    trash=False     # touch tip to each well's edge
```

There is also the new_tip command, which has the same format as the rest of the commands listed, though when using 'never' remember to include a pipette.pick_up_tip() command before the transfer command because it will not pick up one automatically. Also as well as a pipette.drop_tip() if you need to get a fresh tip.

```
pipette.transfer(100,plate.wells('A1'),plate.wells('A2'),
    new_tip='always') #you can set 'always', 'once', or 'never'
```

For the distribute command there is a disposal_vol value that makes the distribute command more accurate. With pipette.min_volume setting a default for the disposal_vol variable. (Which is at minimum 1 uL). Though I believe that the .json that is associated with OT-2 already includes min and max volume of specific pipette tips. This command is primarily used for certain liquids to avoid errors.

```
pipette.min_volume = 20  # `min_volume` is used as default to `
    disposal_vol`
    pipette.distribute(30,plate.wells('A1', 'A2'),plate.rows('2'),
    disposal_vol=10)   # include extra liquid to make dispenses
    more accurate
```

1.2.2 Non-singular Volume Transfer Commands

```
pipette.transfer(
    [20, 40, 60], #volume array
    plate.wells('A1'), plate.wells('B1', 'B2', 'B3'))
```

A volume array gives 20s uL to B1, 40uL to B2, 60uL to B3 Using 0 instead of an actual value will skip the well, which is good for rows or column-based commands.

```
pipette.transfer(
    (100, 30), #volume gradient
    plate.wells('A1'), plate.rows(2))
```

A volume gradient gives 100uL to A2, 90uL to B2, 80uL to C2, . . . 30uL to H2

1.3 Python Building Block Commands

Link to Python Building Block Commands

Building blocks are a lot more complex to work with and are good to us for if we need a very specific procedure. (or if we want to process new equipment/labware) This is linked for you to be able to find it if you really need it, but at this point in time it is not necessary for us to be using building block style commands beyond protocol.delay().

2 GUI Protocol Design

GUI Protocol Design Website

Note: Protocol designer works better when using full screen because the screen does not resize well to smaller sizes and important information gets hidden. This way of protocol creation is still a work in progress on the Opentrons side of creation. The files are exported as a .json and cannot be simulated in current known environments.

2.1 Inital Set-up

This method can be done outside the lab, though does need internet to run the protocol design software as the software is a browser based program. Opening the designer software, you will see (currently) that you are in the file tab with majority of the other tabs greyed out. You can Create New or Import (only json files, which means that you can edit previous protocols made in this program).

When you click Create New a pop-up should come up that focuses on pipette and tip setup, as well as module setup. Once you exit the pop-up of the equipment setup, you need to click on the Liquids tab and click New Liquid. You will need to name the liquid, choose a color, and, if you really want to, describe it. You will need to create at least one liquid before going to design, but I would suggest creating all of the liquids first as doing so will help prevent clicking through tabs (you can pick whatever color you want, just try and do completely different colors not to confuse the liquids you are working with).

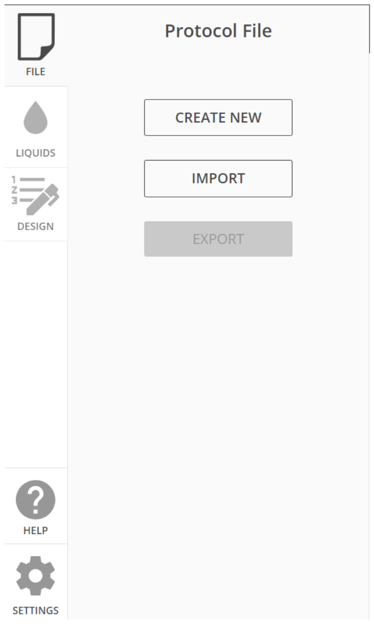


Figure 1: Graphic of file tab

To set up within the Design tab, be on the Starting Deck State and from there you can click on the appropriate slots to add labware. There is a pop up that appears when you click on Add Labware. Our Well Plate is Nest 96 200 uL flat. You will need to include all plates that will be transferred to within the protocol and place all inital liquids before adding steps. Before clicking off of the labware, add a nickname to the labware so it is easier to find. I would suggest slot number-what it does (ex. 7-Soure). When adding liquids to labware you can click and drag to select multiple wells at once, or click on singular wells, and to deselect wells you Shift+Click on wells (can also Shift+Click drag for multiple wells). Though the shift click is only applicable during the initial addition of liquids. When editing liquids you have to select the wells and click the Clear Wells button above the plate view.

You can edit the name and location of the labware at anytime by clicking on the labware in the grid view. To move which slot the labware is in, click and drag the labware to the wanted location. To edit the name/liquid click on the labware and select Name&Liquids. The nickname can be found in the top of the left column.

2.2 Commands

When you click Add Step, a list appears with the majority of the options greyed out (if no modules are selected). With the main steps being Transfer, Mix, and Pause. Pause is a delay for the protocol that can either be set for a certain time, or for the user to resume the protocol via the OT-2 application. Transfer is for the transfer, consolidate, and distribute commands. The basic transfer command moves the Volume Per Well amount from one well to another (the transfer must have the same amount of wells). If you want to control the amount of times the tip is changed during a step, there is a drop down menu (do not select Never as that causes some issues with the protocol).

Distribute can only be selected if you only have ONE source well and multiple destination wells. The volume transferred into the destination wells is selected with the Volume Per Well value. There is another category called disposal volume, just at a minimum have it set to 1 for accuracy reasons. *Note:* Also make sure that you have calculated the liquid necessary for the movement of the protocol because if not all the destination wells of the last aspiration will be messed up.

I recommend having a 5-10% margin of error (ie with 150 uL distributing 9 uL per well having an extra 9/10 uL left in the source well) to be accounted for when aspirating (this should be for all transfer commands, but especially important for the distribute commands). This allotment is not necessary but having that left over amount in the well will help alleviate any possibility of error during setup affecting the final result. With this command you also have the option to blowout the remainder of the final aspiration, either within the Trash or Source Well. This just depends on the sterile level that you want for your protocol, but would say default to Trash.

The consolidate command is the inverse of the distribute command. Which can only be selected with multiple source wells and ONE destination well. Volume Per Well indicates the amount taken from the source wells and put into the destination well.

For all transfer commands there are advanced settings (click the cog wheel), this is primarily done for if there are viscosity differences, which for now don't worry about it, but advice for that will be included in "Aspiration Changes for Different Liquids" section.

The mix command focuses on mixing a certain volume of liquid, a certain number of times for the selected wells. The mix volume can only be as much as the max volume of the pipette tip.

2.3 Export File

When you want to check the Final Deck State, scroll to the bottom of the Protocol Timeline. *Note:* All steps can be minimized with the arrow on the right side of the step name.

To export go back to the File tab and click the now dark grey Export file button. There will be a pop up to say, "hey this can only be run on this version" OPT can run it from the last I checked because I updated OPT before I left. Exact message: "This protocol can only run on app and robot server version 6.2 or higher. Please ensure your OT-2 is updated to the correct version."

2.4 GUI Module Setup

Note: We have temperature module GEN2. Thermocycler should be GEN 1 if I am not mistaken (please check before actually running it please)

2.4.1 Module Set-up

For the modules, you have to Create New file and select the modules in the beginning file page. *Note:* Since the tip rack is auto-set to slot 1 in order to place any of the modules in slot 1, you need to go to the design tab and move the tip rack to another location and then you can place a module in slot 1. My personal preference is temp module slot 1 as it is easier to wire that way.

In order to actually place liquid into the module, you will need to define the labware of the modules by clicking on the dashed location on the module and clicking add labware. The thermo module only has 1 option for the module. Recommendation do not have liquid starting in the thermo module because it is a pain to set up that way.

The temp module uses either the 24 well or 96 well block with inserted capsules. This varies on what you want to do with the system and makes it easy to take out what has been worked on.

2.4.2 Module Commands

When you click on the temperature step, you can set the temperature of the module (seen as the nickname set), or deactivate the module (which you need to do at the end of the protocol). The program will ask if you want to place a pause if you set the module to a certain temperature. I would recommend so. The temperature module has a temperature range of (4-99 °C)

The thermocycler step will pop up with deactivated lid and block, if you flip the toggle switch, you can set the temperature for the thermo module (lid 37-110 °C and block 4-99 °C). Unlike the temp module, the thermo module does not have a pause until temperature is met, so if you need the block to be at a certain temp I would recommend heating the thermo and temp module simultaneously if there needs to be a constant temperature to the sample and absolutely no change in temperature.

Though if you select Program a Thermocycler Profile you can select volume in wells, lid temp, and for each step (single time) or cycles (multiple times), you select block temperature and amount of time moving the samples. This will heat up the thermocycler and hold at that temperature during the process, but afterwards you will need to select if the Thermo module holds that temperature or is deactivated (which will cool/warm the sample). If you need to grab the sample later from the thermocycler I would recommend moving the sample back to the temp module after previously deactivating the thermocycler. The temp and thermo modules can be turned off not in program, but the temp module is easier to deal with if a sample is needed to be kept at temperature for a while and someone cannot be there right at the end of the protocol (also moving liquid in/out of thermo module by hand is not fun).

Note: If you have programmed OPT to transfer in/out of the thermo module with the lid closed you will get a fancy red bar that says “The thermocycler lid is closed” so you will need to open the lid in a previous step or add a step (Change Thermocycler State) and click open lid (it is easier to edit previous step, but you do you).

3 Module Setup

Warning: Remember to deactivate all temperature-based modules at the end of code because it will stay at requested temperature while powered.

Note: Both the thermo and temp module can be used simultaneously. Pay attention to placement of labware around the modules as it get a little crowded with how the modules take up space versus the labware (especially the thermo module).

3.1 Temperature (temp) Module

The temp module can be placed pretty much wherever, but to standardize it use grid 1 for the module location, primarily as that is the best location for cords and module space usage if both thermo and temp modules are used. We have 2 metal well plates that can be used in the temp module: the 96 and 24 well plate.

3.1.1 Physical setup

To physically set up the temperature module, you will need to take off the lower left and lower back magnetic panel (you won't break it, it's just stubborn) and then set the temperature module with exhaust facing outwards. You will need to use the power adapter to plug into the wall and the data cord, which should be in the box with the module box *Note: I could be wrong, and the power adapter is near the pipette mounts.* My suggestion for the data cord is to not run it through the machine (avoids accidental collisions), but outside and around to the back to plug into the black data port array in the top back of the machine (you should be able to use any one of the USB-A ports available, I have yet to find an issue with using any of those). Power on the device with the side button.

If you have the protocol setup open on the OT-2 app it should let you know if it can detect the module. If not make sure to run through again to make sure that the module is properly connected.

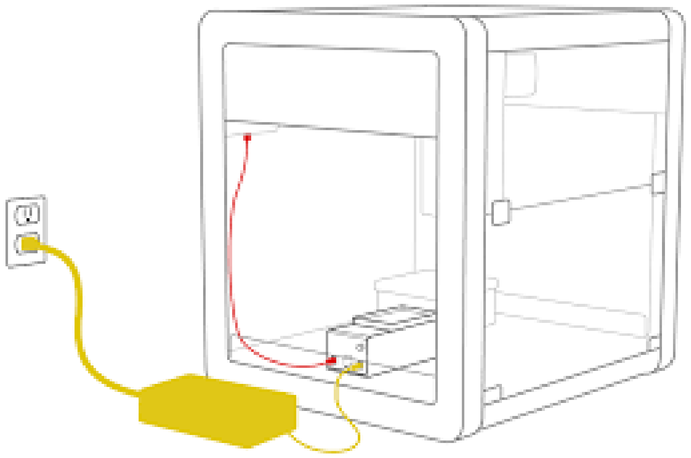


Figure 2: Graphic of temperature module wiring

3.1.2 Python Setup

When setting up the software you will need to have already defined the metadata as well as define the protocol

```
temp_mod = protocol.load_module('temperature module gen2', '1')
plate = temp_mod.load_labware('corning_96_wellplate_360ul_flat')

temp_mod.set_temperature(celsius=4)

temp_mod.status  # 'holding at target'
protocol.delay(seconds=20)
#can do minutes=value (cannot do hours)
temp_mod.deactivate()
temp_mod.status  # 'idle'
#tempurature range is 4-95 degrees C
```

3.2 Thermocycler Module

3.2.1 Physical Setup

To physically set up the thermo module, it is pretty similar to the temp module, but with the added benefit that the module is heck of a lot heavier, and the power adapter is in its own box that sits above the thermo module box. The thermo module takes up grid slot's 7 and 10, and need to have both back AND left panel taken off for it to be wired and fit. Be sure to double check that the module lines up correctly and gets locked into the grid as the module will not be able to transfer liquid correctly if misaligned (it is 2.5 grids long so the underneath gridlines might be mismatched).

The power adapter and data cord need to be plugged into appropriate outlets. The thermo's power adapter switch is the power switch for the thermo module while temp module has a button on the side of the module. The button on top of the thermo module is to open/close the thermo lid out of protocol. Make sure to screw on the power adapter cord to the module, since it prevents accidental power loss if the cord gets shaken loose. The data cord should be wound around to the backside of the machine and wound up alongside the exhaust pipe to avoid accidental interaction with the pip mount, or heat from exhaust.

3.2.2 Python Setup

I will say that writing your own code for the thermocycler is probably not the best move, since there are quite a few parts. I would recommend modifying the thermo module code found on the github. If you need help modifying the thermocycler code reach out and tell me what you need to do because it is stupidly tedious to get correct with all the moving parts of the thermo module. Also, if you need help I might be able to

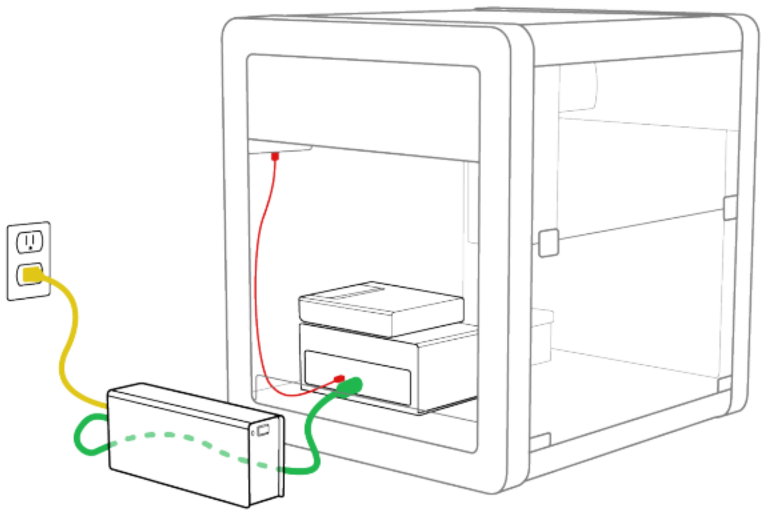


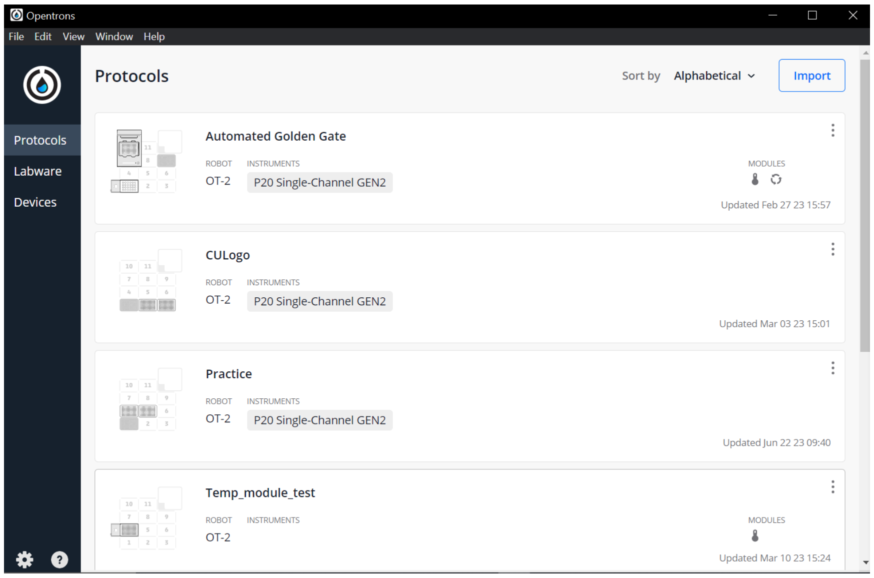
Figure 3: Graphic of thermocycler module wiring

create better protocols for specific uses so that modifications are faster for common protocols.

4 OT-2 App to Machine Interaction

This tends to be somewhat self-explanatory once you have the OT-2 app downloaded and updated to the proper version. *Note: if OPT needs to be updated I will handle that so do not worry about that.* Once you flip the on/off switch on the back left of the machine towards the bottom and press the circular button to turn OPT on. You can plug into OPT using the USB-A before turning OPT on, but it does take a bit for OPT to be properly connect to the OT-2 app. If OPT is not registering as connected, unplug and re-plug the USB-A connector as sometimes it takes a bit to register within the application (These instructions are subject to change if we get a proper lab computer for OPT).

Inside the OT-2 app you will see on the left: protocols, labware, and device. Within the protocol tab you will see an import protocol button (top right) as well as a list of already imported protocols sorted alphabetically *Note: You can change how it is sorted, but I recommend alphabetical.* (If we have a lab computer that is not properly set up to the internet, there will be a flash drive connected to the lab computer so that people can transfer files to the lab device, since no one carries one anymore). The Labware tab is not to be messed with. The Device tab shows what devices are available to connect as well as unavailable devices. Connected device information can be accessed through the device tab: wired IP, device number, calibration test, etc.



Protocols

Automated Golden Gate
ROBOT OT-2
INSTRUMENTS P20 Single-Channel GEN2
MODULES
Updated Feb 27 23 15:57

CULogo
ROBOT OT-2
INSTRUMENTS P20 Single-Channel GEN2
Updated Mar 03 23 15:01

Practice
ROBOT OT-2
INSTRUMENTS P20 Single-Channel GEN2
Updated Jun 22 23 09:40

Temp_module_test
ROBOT OT-2
INSTRUMENTS
MODULES
Updated Mar 10 23 15:24

Figure 4: Graphic of protocol tab

To run a protocol, click on the Protocol tab and it will open a robot configuration page with a blue start setup button in the corner. OPT will let you know if the pipette-mount or modules are not registered as connected. So double check that the pipette-mount is the correct size, volume, or number, or if the modules are powered, plugged-in, or turned on.

Warning: make sure that the labware is set up correctly because OPT cannot sense whether you placed them correctly. Though can tell if a pipette mount is empty.

Before running a protocol, I would recommend simulating the protocol to make sure that you know where and what is happening to your protocol. This makes the possibility of wasted material and errors to go down. I will try and get a simulation terminal set up on whatever computer is connected to OPT so if you need to adjust the protocol for whatever reason you can simulate in the lab without needing to leave/use personal computer.

5 Simulation Setup and Usage

Note: I personally use Ubuntu 20.4 since the Linux style terminal is easier than Windows terminals for Opentrons. The process for other OS is similar and if you need to reference the Command Line Simulation Article. After installation and setup, internet is not needed to run simulation.

Install python 3.7 onto your device, from there it is mainly in Ubuntu, or other terminal depending on OS.

Python straight command *Note: might need to do pip3 and not pip*

```
pip install opentrons
```

```
pip install -upgrade opentrons
```

Note: This is for double checking the simulation is up to date

Get to your folder where protocol.py is. To get from Ubuntu files for windows files use cd ../.. And the cd mnt to escape from the ubuntu files and then cd c/.../... to the folder desired. *Note:* This idea follows same idea for Mac users, but not sure how to file find in Macs since I haven't used one in years. I will say that it seems that Mac terminals might be able to run this easier than Windows, but I have no idea.

To run the simulation use

```
opentrons_simulate protocol.py
```

Or

```
opentrons_simulate -e protocol.py
```

Or

```
opentrons_simulate -o runlog protocol.py
```

-e is estimation for how long the protocol will take to run *Note:* not very accurate for modules (and some transfer commands) and gives the same instruction line simulation as -o runlog.

Warning: GUI protocol design cannot be simulated through this version of simulator as the simulator is an older version of the OT-2 App/-Machine interface than what the GUI protocol currently uses. Also the GUI outputs a .json instead of a .py script.

6 Appendix

6.1 Aspiration Changes for Different Liquids

Link to Viscosity Paper

It is important to modify the protocol for different viscosities of liquid, for all liquid automation machines. If the aspiration rate is off the error of the protocol increases by quite a bit because of the whole wrong volume being moved thing.

There are plenty of ways to handle this step, but unless you are working with some really odd liquids this should not be an issues. Though if you have any questions reach out to me, I will have done plenty of comparisons on the methods, which includes use of aspiration rates, blow-out, touch-tip, and delay commands. I did link a paper on some examples on how to deal with different pipettes, liquids, and handling styles from Opentrons.

6.2 Some Errors in Simulation Defined

- telling you that a slot on the device is currently being used (move stuff)
- error on a specific line (various reasons, usually has other indicator in the simulation information)
- protocol is not defined (i.e. hierarchy tabbing is not good, usually)
- unexpected tabbing (you tabbed too much)
- TypeError(loc): #, this is a formatting issue that usually is .wells/ .rows/ .columns command syntax errors