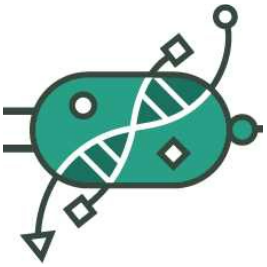
Jan 16, 2024 Version 1

OT-2 Modular Cloning Construct Assembly V.1

DOI
dx.doi.org/10.17504/protocols.io.5jyl8p82rg2w/v1



Ana Mariya Anhel[1], Laura Cutugno[1], Ángel Goñi-Moreno[1]

[1]Centro de Biotecnología y Genómica de Plantas, Universidad Politécnica de Madrid (UPM)-Instituto Nacional de Investigación y Tecnología Agraria y Alimentaria (INIA/CSIC), Madrid, Spain

Biocomputation Lab
CNB-CSIC

DOI: **https://dx.doi.org/10.17504/protocols.io.5jyl8p82rg2w/v1**

Protocol Citation: Ana Mariya Anhel, Laura Cutugno, Ángel Goñi-Moreno 2024. OT-2 Modular Cloning Construct Assembly . **protocols.io** **https://dx.doi.org/10.17504/protocols.io.5jyl8p82rg2w/v1**

Protocol status: Working
We use this protocol and it's working

Created: December 14, 2023

Last Modified: January 17, 2024

Protocol Integer ID: 92328

Keywords: modular cloning construct assembly, modular cloning, modular cloning construct, modular cloning constructs from different part, golden standard modular cloning of level, python script for an opentron, description of the lap repository entry lap, lap repository entry lap, assembly of the construct, assembly, plasmid, ot2, set of instruction, laboratory, opentron

Funders Acknowledgements:
Comunidad de Madrid
Grant ID: Y2020/TCS-6555, 2019-T1/BIO-14053
MCIN/AEI
Grant ID: CEX2020-000999-S, PID2020-117205GA-I00
European Research Council
Grant ID: 101044360

Abstract

This protocol is meant to create modular cloning constructs from different parts into a final plate and, optionally, perform the temperature profile needed to the assembly of the constructs.
The output of running this script will be the final plate(s) with the constructs and the mix needed to perform that assembly, and the corresponding map(s) with the names of the constructs in their corrresponding well which will be given by the user in the input file.

This protocol uses a python script for an Opentrons 2 robot and an excel file containing the required variables to set the number of samples, volumes of transfer, type of plates, etc...

In our laboratory, this protocol has been used to perform plasmids using the Golden Standard modular cloning of levels 1 and level 2

This protocol is a set of instructions or description of the LAP repository entry **LAP-MoCloAssembly-OT2-1.0.1**

Guidelines

This protocol was developed with python 3.7.1, OT App Software Version 6.3.1 and API level version 2.14 in a Linux 4.14.74 system (these are the OT-2 specifications).
In the script several packages are used: pandas (0.25.3), openpyxl (3.1.2), math, random and numpy (1.15.1)

This protocol has been tested by assembling level 1 and level 2 constructs with parts of the golden standard database

Materials

Software

- Python 3.7.1
- opentrons software version 6.3.1
- python packages: pandas (0.25.3), numpy (1.15.1), openpyxl (3.1.2), math, random
- OT App
- Excel

OT-2 Labware

- Opentrons Tip racks

Equipment	
Opentrons 96 Tip Rack 300 µL	NAME
Tip rack	TYPE
Opentrons	BRAND
-	SKU
https://labware.opentrons.com/opentrons_96_tiprack_300ul?category=tipRack	LINK

Equipment	
Opentrons 96 Tip Rack 20 µL	NAME
Tip rack	TYPE
Opentrons	BRAND
-	SKU
https://labware.opentrons.com/opentrons_96_tiprack_20ul?category=tipRack	LINK

- PCR skirted plate

Equipment

PCR 96-plate low profile Thermo Scientific	NAME
PCR Plate, 96-well, low profile	TYPE
Thermo Scientific	BRAND
AB0800R	SKU
https://www.thermofisher.com/order/catalog/product/AB0800R?SID=srch-srp-AB0800R	LINK

- Opentrons Eppendord Tube Rack

Equipment

Opentrons 24 Tube Rack with Eppendorf 1.5 mL Safe-Lock Snapcap	NAME
Tube Rack	TYPE
Opentrons	BRAND
opentrons_24_tuberack_eppendorf_1.5ml_safelock_sna	SKU
https://labware.opentrons.com/opentrons_24_tuberack_eppendorf_1.5ml_safelock_snapcap?category=tubeRack	LINK

- Tube Rack Eppendorf for Heater-Shaker + 24 eppendorf tube holder

TubeHolder_Eppendorfs_HS_OT2.stl

Equipment	
24 Eppendorf Tube holder	NAME
Tube holder	TYPE
Opentrons	BRAND
999-00030	SKU
https://shop.opentrons.com/4-in-1-tube-rack-set/	LINK

- 1.5mL eppendorfs (without the cap) + 4ºC cold-block with adaptor (file attach)

Equipment	
BRAND™ Centrifuge Tube Mini-Cooler	NAME
ColdBlock	TYPE
BRAND	BRAND
10141921	SKU
https://www.fishersci.es/shop/products/brandtech-scientific-brand-centrifuge-tube-mini-cooler-3/10141921	LINK

adaptor_OT_coldblock.stl

Reactives:

- **Water:** MilliQ water

- **T4 Ligase and Ligase Buffer:** T4 DNA Ligase, 100u **Promega** **Catalog #**M1801
- **Restriction Enzyme:** BpiI (BbsI) (10 U/µL) **Thermo Fisher** **Catalog #**ER1012
  BsaI-HFv2 **New England Biolabs** **Catalog #** R3733S

Equipment:

Equipment	
OT-2	NAME
Liquid handler	TYPE
Opentrons	BRAND
OT-2	SKU

Equipment	
Single Channel Electronic Pipette (GEN2) 300uL	NAME
Opentrons Pipette	TYPE
Opentrons	BRAND
-	SKU
https://shop.opentrons.com/single-channel-electronic-pipette-p20/	LINK

Equipment

Single Channel Electronic Pipette (GEN2) 20uL	NAME
Opentrons Pipette	TYPE
Opentrons	BRAND
-	SKU
https://shop.opentrons.com/single-channel-electronic-pipette-p20/	LINK

Equipment

Opentrons Thermocycler Module	NAME
Thermocycler	TYPE
Opentons	BRAND
999-00174	SKU
https://shop.opentrons.com/thermocycler-module-1/	LINK

Equipment	
Opentrons Heater-Shaker Module	NAME
Heater-Shaker	TYPE
Opentrons	BRAND
999-00157	SKU
https://shop.opentrons.com/heater-shaker-module/	LINK
	

Troubleshooting

Safety warnings

❗ If you are using the heater-shaker take in account that there is a limit of RPM that it can shake before the liquid of the eppendorfs get out. As well, take in account that there are constrictions that could prevent some labware to be placed.
This speed depends on the liquid consistency and volume

Before start

Being a one-pot reaction in this protocol only 1 restriction enzyme is used so one type of level, in the case of the golden standard database, can be produced in only 1 run

Files Preparation

1 **Preparing Customized Template**

Preparing the template (a .xlsx) with the specific variables for each experiment.

Here there is attached a template of the variable file with several sheets and a PDF file explaining each variable:

1. **GeneralVariables:** variables related mainly to the labware that is going to be used
2. **PerPlateVariables:** variables related to the specifications of each source plate
3. **PipetteVariables:** variables related to the pipettes that are going to be used
4. **ReactionVariables:** variables that will determine the final mix of the wells
5. **ModuleVariables:** variables related to the modules used in the protocol, the thermocycler and the heater-shaker
6. **Combinations:** set of combinations that are going to be created in the final wells, one combination per row and one DNA part per cell
7. **Map DNA Parts Sheet(s):** sheet(s) with the names of each DNA part that can be used to create final assemblies denoted in the combinations sheet. The sheet(s) need to have also dthe name of the rows and columns of the plate and the wells that does not have any sample need to be left empty → *not included in the template but needed to be included and have the same names as established in the variable **Name Map DNA Parts** from the PerPlateVariables Sheet*
8. **TemperatureProfile (Optional):** a profile that will be performed in the thermocycler if set as True in the ModuleVariables sheet

XLSX Template-VariablesMoCloAssembly...

PDF MoCloAssemblyInstructions.pdf

1.1 *Fill the template with the corresponding values*

1.2 *Store it with the name VariablesMoCloAssembly.xlsx*

Note

The file should be spelt **exactly** *VariablesMoCloAssembly.xlsx* or the Python script won't work correctly

2 Transferring file to Robot

Transfer the *VariablesMoCloAssembly.xlsx* to the directory */data/user_storage* of the OT robot
that we are going to use to perform the protocol.

Note

Before transferring any file to the OT, we need to know the **IP of the robot**.

This can be obtained in the Networking section of the Device that is going to be used.
To obtain this info go to **OT-App -> Devices -> Chosen Robot (three dots) -> Robot Settings -> Networking**

In this tab, you can see 2 types of IP; one is shown if both the robot and you are connected to the same WiFi, and the other is shown if the computer and the robot are connected via USB. Both connections can be used for this step.

Note

To connect to the robot an **OT-key** should have been previously generated, and it is done with the *ssh-keygen* command and **transferring the public key to the OT.**

For more information about how to generate and set the connection between your computer and the Opentrons robot, visit https://support.opentrons.com/s/article/Setting-up-SSH-access-to-your-OT-2

Here, we present a summary of how to transfer the files in 3 Operative Systems: *Windows, MacOS* and *Linux*

MacOS/Linux
We will use the command line with scp to transfer the file *VariablesMoCloAssembly.xlsx* to the OT system.

We need to perform the following command

Command

File passing from our computer to robot's linux (OT raspberry)

Passing Files to OT

```
scp -i [ot_key] [file] root@[IP_OT]:/data/user_storage
```

Note

You could face difficulties transferring files in MacOS Ventura (13) and Sonoma (14). These problems can be solved by adding the argument -O (uppercase o) to the command

Command

Transferring files to OT (MacOS 13 and 14)

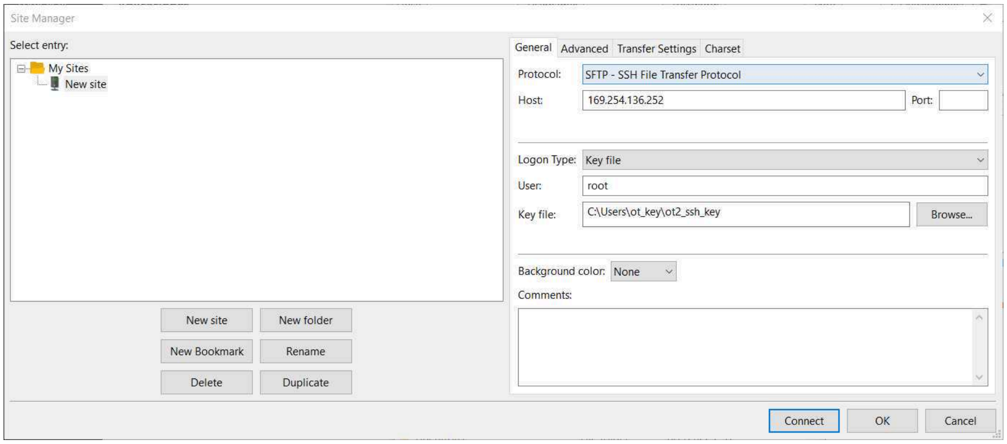
Transferring files to OT

```
scp -Oi [ot_keypath] [file path]
root@[OP_robot]:/data/user_storage
```

Windows
There are several ways to send files from a Windows to a Linux (for example, with a virtual machine or Windows Powershell in the latest versions of Windows).

Here, we will use **FileZilla** (https://filezilla-project.org/download.php?type=client).

Go to **File → Site Manager → New Site → Change Protocol to SFTP**. Then, introduce in Host the OT IP, change the *Logon Type* to key file, change *User* to root and give the directory where the ot key is. It should look something like this



Protocol: SFTP - SSH File Transfer Protocol
Host: 169.254.136.252
Port:
Logon Type: Key file
User: root
Key file: C:\Users\ot_key\ot2_ssh_key
Background color: None
Comments:

Example of setting the FileZilla to transfer files from Windows (our computer) to Linux (OT)

Then press *Connect*, and we will have a connection between our computer and the robot.

After this connection, we should be able to move the file *VariablesMoCloAssembly.xlsx* (in our computer) to the directory */data/user_storage* in the robot.

This method can be used as well in some Linux and MacOS

Note

Take into account that the IP of the robot could change, so it is possible that it will be needed to change the host in these connections from time to time.

3 Adding the custom labware

There is only a need to do this step when the labware that you are using is not OT official or is not included in the OT app

3.1 *Creation of .json file*

The description file can be obtained by describing the labware dimensions at **https://labware.opentrons.com/create/**

3.2 *Uploading files to the OT App*

In the OT app, we need to perform the following route: **Labware → Import → Choose File → Select file we have created in step 3.1**

Expected result

After uploading the labware you should be able to see the new labware in the Labware tab of the OT App, all custom labware can be found more easily in the category *Custom Labware*

3.3 *Transfer labware files to the robot*

If you are using the entry **LAP-MoCloAssembly-OT2-1.0.0 or LAP-MoCloAssembly-OT2-1.0.1 with custom labware,** an additional step is needed, which is transferring a folder with the custom labware

We need to create for our custom labware a folder with the API name containing the description file (.json) called 1.json and then transfer that folder to the robot's folder */data/labware/v2/custom_definitions/custom_beta* in a similar way as in the Step 2 but with the difference that is a directory that needs to be transferred and not a file.

Command

Transferring the custom labware to OT (Linux)

Transferring the custom labware to OT

```
scp -i [ot_key] -r [directory_custom_labware] 
root@[IP_OT]:/data/labware/v2/custom_definitions/custom_beta
```

Note

We do not need to execute this part every time the protocol is used, only when that labware is not included in the OT official labware and these directories are not in the robot

Prepare RobotOS

4 **Install needed packages**

This script needs the package *openpyxl*, which is not installed by default in the OT-2 robots

Note

This step is only needed if the package is not installed in the robot, not every run of the protocol

If the package is not installed, an error will appear when running the script in the robot. While simulating the script in the app, this error will appear, but you can ignore it

4.1 *Connect to the robot*

go to step #2 to find the IP of the robot in which you want to run the script

To connect to the robot, you can do it via ssh with the following command

Command

Connect to Linux based OT via ssh

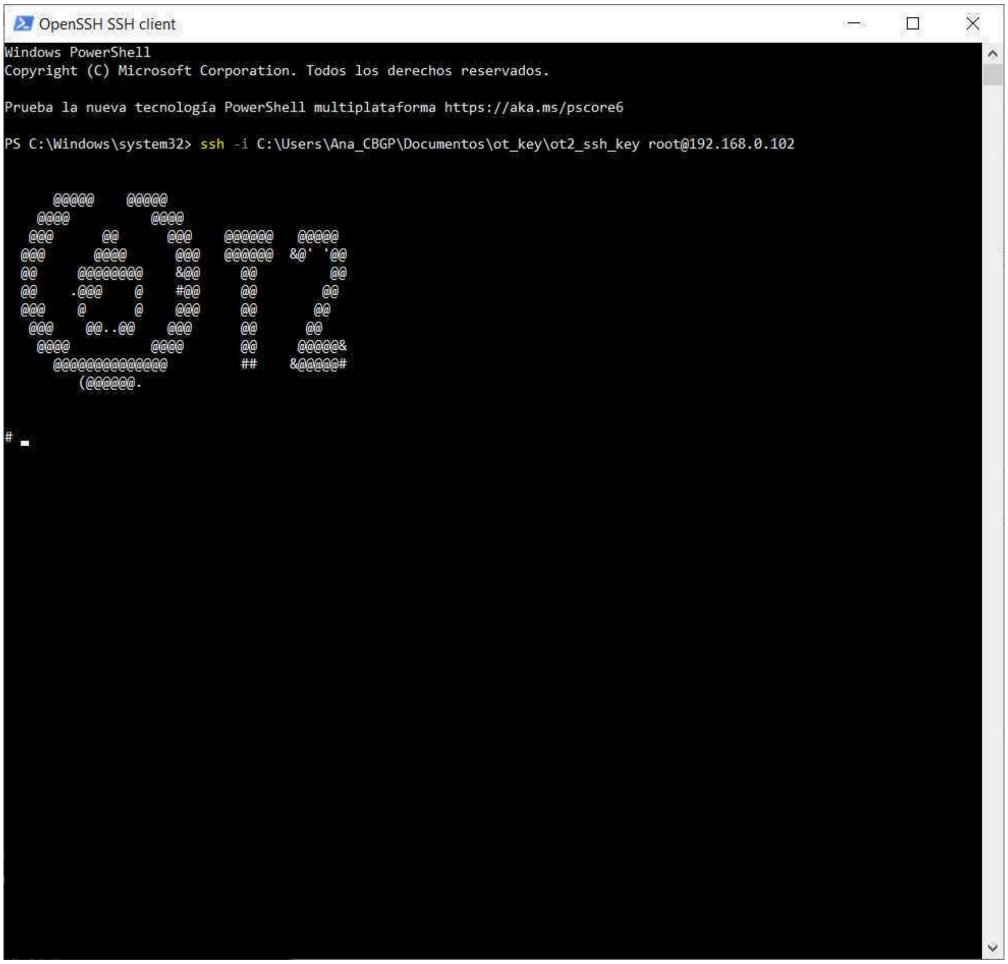
Connect to Linux based OT via ssh

```
ssh -i [path ot_key] root@[Robot_IP]
```

In Windows you can do this command in Windows Powershell

Expected result

If the connection has been successful you should obtain a screen similar to the following image



```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\Windows\system32> ssh -i C:\Users\Ana_CBGP\Documentos\ot_key\ot2_ssh_key root@192.168.0.102


      @@@@@    @@@@@
    @@@@          @@@@
   @@@      @@      @@@    @@@@@@   @@@@@
  @@@      @@@@      @@@   @@@@@@  &@' '@@
  @@     @@@@@@@@    &@@     @@         @@
  @@    .@@@    @    #@@     @@        @@
  @@@    @      @    @@@     @@       @@
   @@@    @@..@@    @@@      @@      @@
    @@@@          @@@@       @@     @@@@@&
      @@@@@@@@@@@@@@         ##    &@@@@@#
         (@@@@@@.


#
```

Drawing obtained when entering an OT-2 system

4.2 *Install the package*

Once inside the robot's system, you need to run the following command

Command

Install openpyxl package (Linux 4.14.74-v7)

Install openpyxl package

```
pip install openpyxl
```

Note

For more information about installing packages in the opentrons robots, check the following Opentrons page: https://support.opentrons.com/s/article/Using-Python-packages-in-Python-API-protocols

Running Protocol

5 **Load script in OT-App**

Now that we have transferred the variable files to the robot, we can load the script and run it in the selected robot

Note

This whole step has been developed with version 6.3.1 of the OT-App and has been tested until version 7.0.2

Indications may vary from version to version

Software

Opentrons App	NAME
Windows >=10, Mac >=10 , Ubuntu >=12.04	OS
Opentrons	DEVELOPER
https://opentrons.com/ot-app/	SOURCE LINK

5.1 *Load the script in the App*

Protocols → Import → Drag Python script

Note

The last script version can be found at **https://github.com/BiocomputationLab/LAPrepository/tree/main/LAPEntries** (the name of this file is the user's choice). The name of the directory should be **LAP-MoCloAssembly-OT2** followed by the version.

As well we can find the latest version of the script at **https://www.laprepo.cbgp.upm.es/repository/** with the same name as in GitHub

Software

LAP Repository	NAME
https://biocomputationlab.com/	DEVELOPER
www.laprepo.com	SOURCE LINK

Note

The App with version 6.3.1 analyzes your protocol before setting a robot to run, so the labware will not be shown before assigning the protocol to a specific robot when you import it into the App

5.2 *Select Robot to Perform Script*

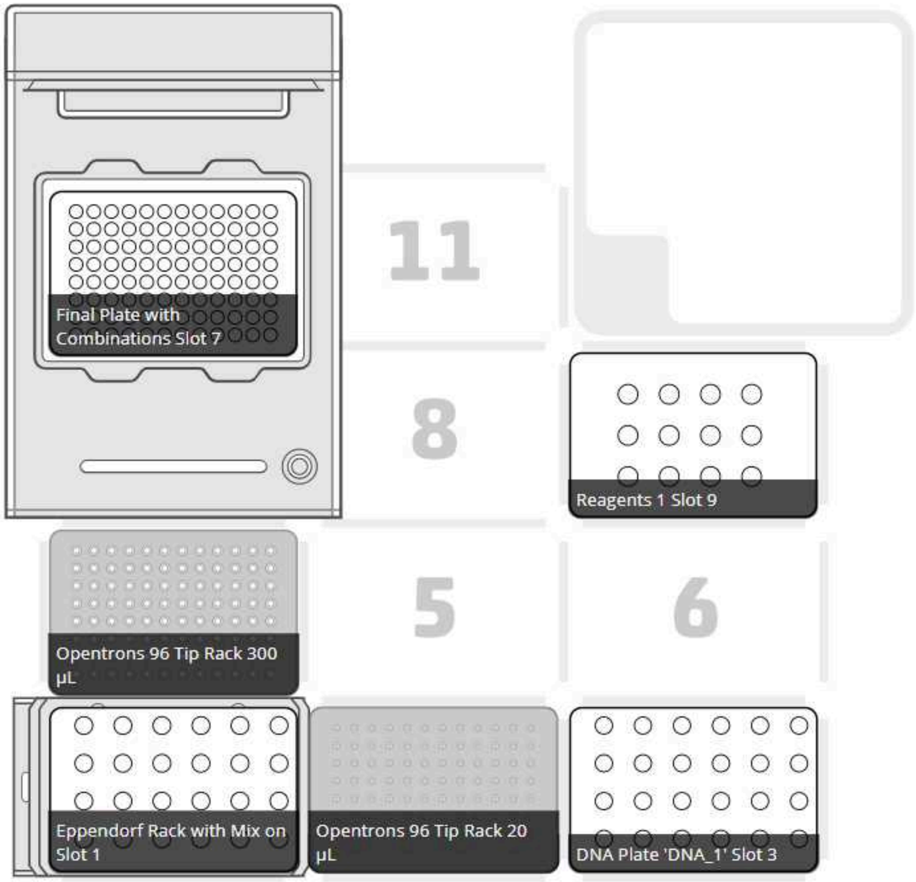
Click in the protocol → Start setup → Choose the OT where the file *VariablesMoCloAssembly.xlsx* is → Proceed To Setup

After clicking on Proceed to Setup, you should obtain, if there is no error, the positions of the labware in the *Labware* tab and the reagents, with their corresponding volume in the *Liquids* tab.

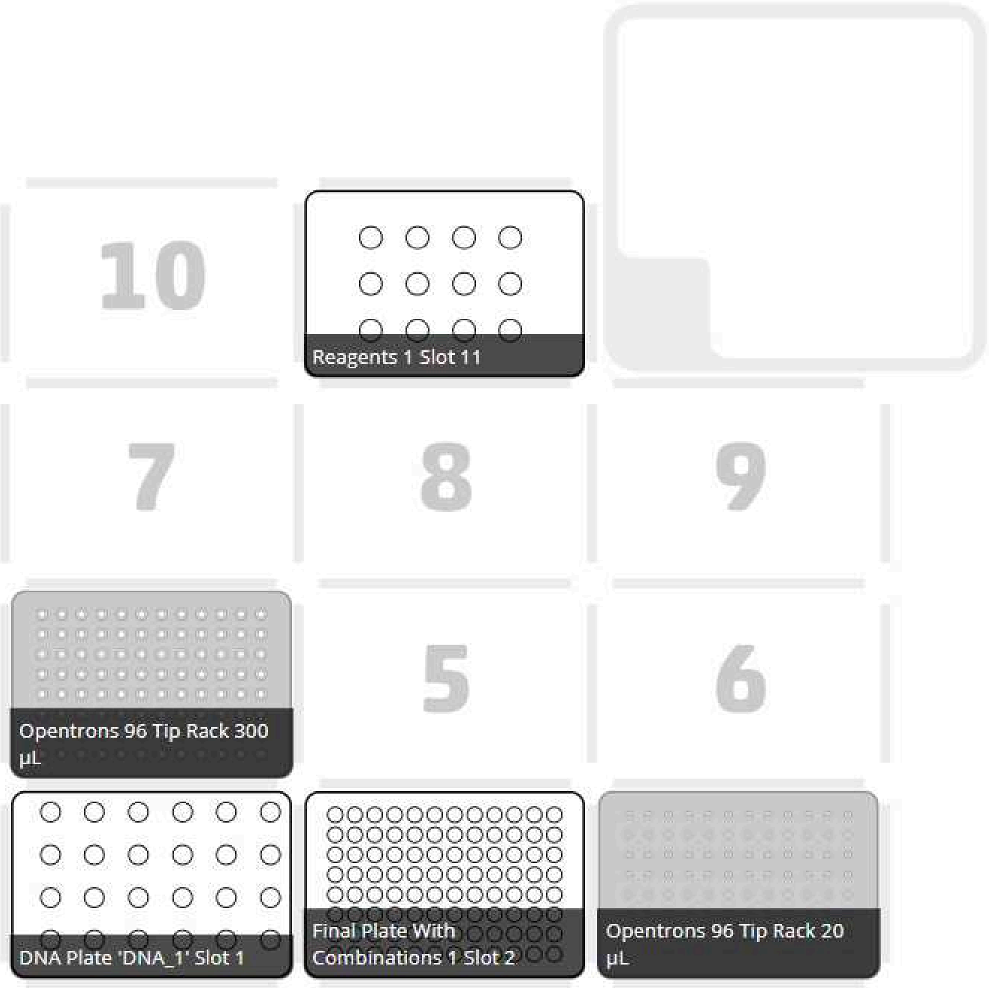
In case the protocol with the set variables cannot run, an error will be raised by the app. Many errors are contemplated already and have a specific message that will give the user a hint of what could have gone wrong.

Expected result

A labware setup should look like the following image, where you can find the initial and final plates, the Eppendorf labware to store the reagents, the corresponding tips and, if included, the location of the heater-shaker(s) and thermocycler. The latter will always have the exact location in the OT-2



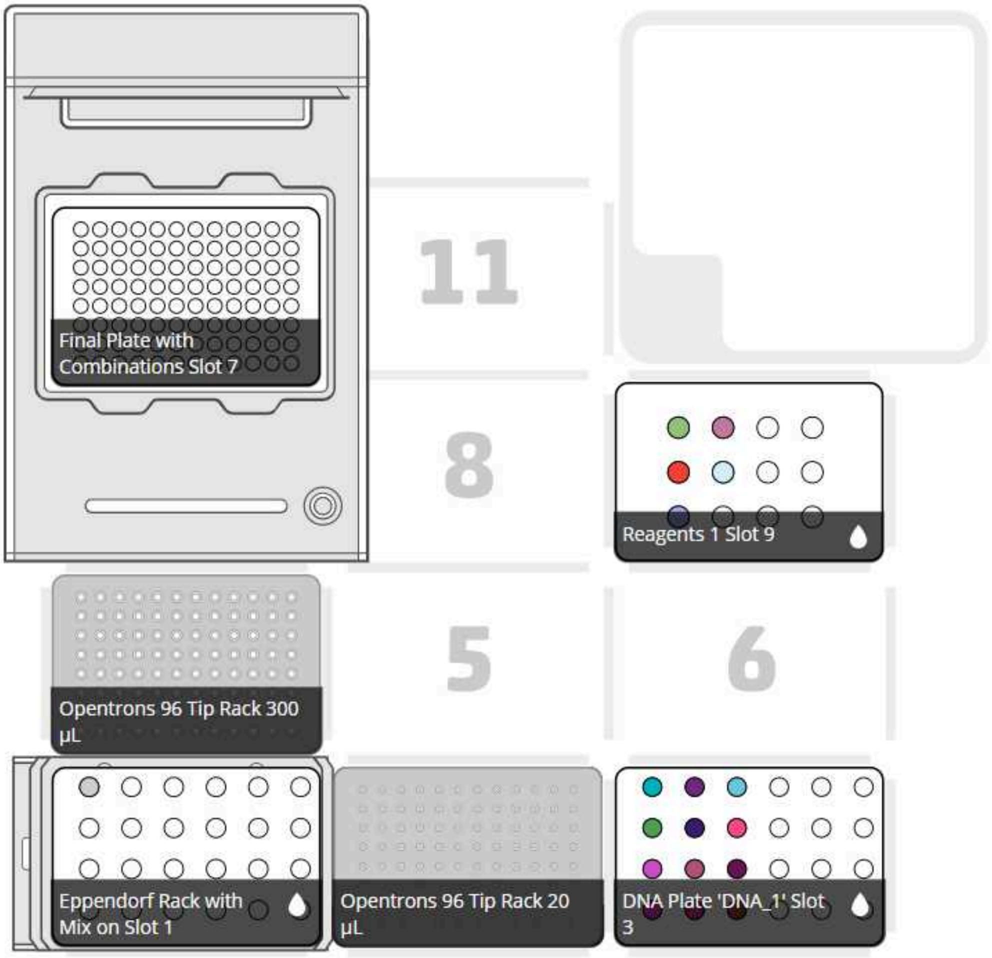
Labware Set-up example of a MoClo assembly protocol where both modules are set as True



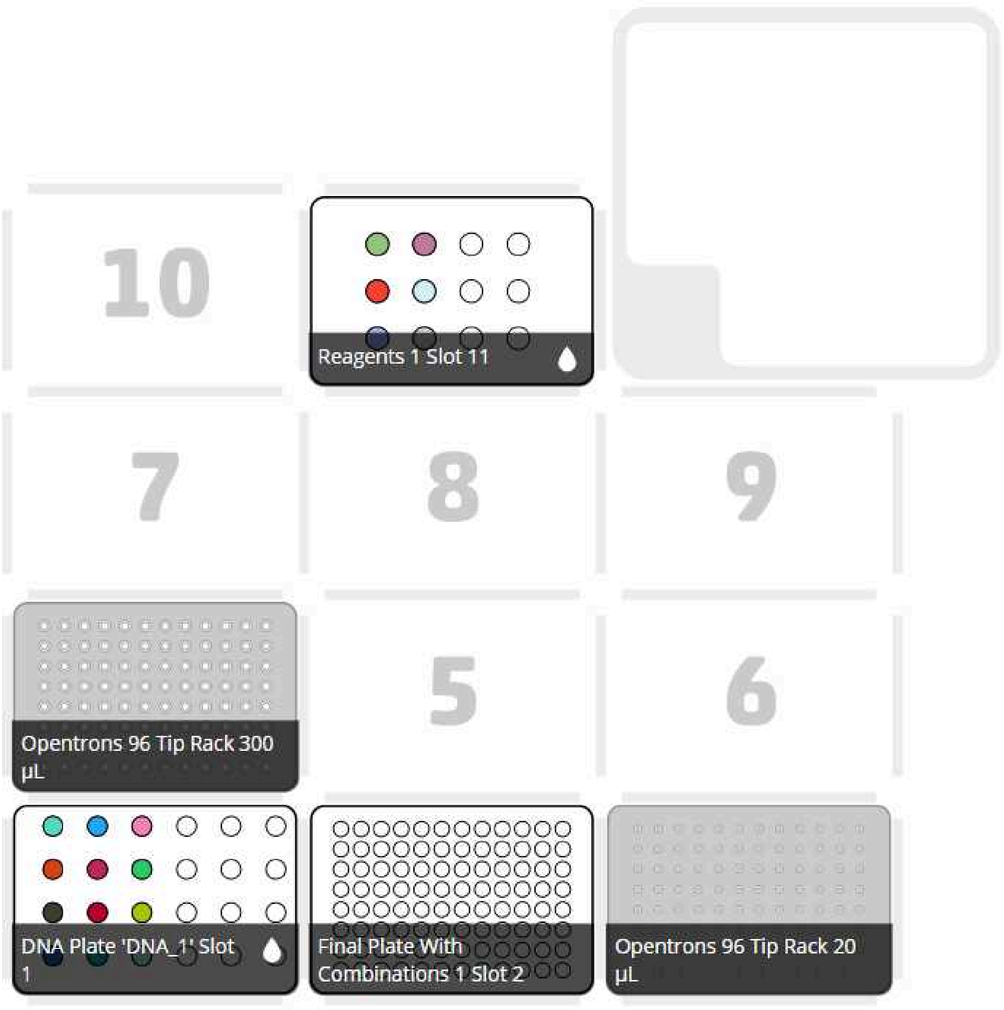
Labware Set-up example of a MoClo assembly Protocol where both modules are set as False

Expected result

A liquid setup should look like the following images, where you can find the samples in the initial plates and the different reagents in the Eppendorf labware with their respective volume. You have the reagent labware(s) where you can find the volumes and positions of the enzymes, ligases, water, etc. The you have the DNA plate(s) where you can find the different parts you have inserted in the input variables file and their needed volume



Liquid Set-up Example of a MoClo assembly Protocol where both modules are set as True



Liquid Set-up Example of a MoClo assembly Protocol where both modules are set as False

Note

Both the volume of the reagents and the DNA parts are exactly what is needed, so it is **suggested to pour always more to take in account the error of pipetting**

Note

It is recommended that you perform a labware position check.

You can do it with test plates and tube racks after loading the script but before cleaning the surface. That way, you reduce the probability of contamination (using the test plates and labware) and pipetting errors (position check).

6 Run Protocol in OT

6.1 *Make sure the needed calibrations are done*

Pipettes, tip racks and tip length calibrations need to be done for the items used in this run

6.2 *Labware position check is performed (if needed)*

6.3 *Clean the surface of the robot with 70% ethanol to clean and disinfect the surfaces*

Note

Check the Opentrons page https://support.opentrons.com/s/article/Cleaning-your-OT-2? for more information about cleaning the OT-2 robot with the proper materials.

6.4 *Set the labware and reagents as shown in the OT-App*

6.5 *Start Run*

The procedure that the robot is going to do is mainly divided into 9 parts:

1. (Optional) The block temperature from the thermocycler reaches a temperature set by the user in case that variable is not empty
2. Distribute the needed water to each final well (the volume of water is calculated by the OT-2 according to the volume set in the variable *Volume Final Each Reaction (uL)* taking in account the volume of the other reagents and the number of DNA parts that is going to that specific well)

3. Creation of mix(es) transferring ligases, buffer, serum and restriction enzyme to new tube(s)
4. Mixing with either a pipette or heater-shaker
5. Distribute mix to final plate(s)
6. Distribute DNA parts to the corresponding wells (as many transferrings as set in the combinations sheet)
7. Generate identity maps to be exported
8. (Optional) Pause the program for user to put lids on the plate located in the thermocycler (if in the input variable)
9. (Optional) Temperature profile with thermocycler module
10. (Optional) Block of thermocycler stay to the set temperature

Expected result

One or more plates where there is a mix between the different DNA parts and the mix of reagents needed to perform the assembly process, each combination will be in one well.

A **sheet for every final plate will be created as well in an Excel file** with the given name in the sheet GeneralVariables in the variable "Name File Final Constructs" followed by the extension .xlsx in the folder */data/user_storage* of the robot where we run the script.

After-Running

7 **Retrieve labware from the OT**

8 **Importing map from robot**

To retrieve the file we can go to step #2 and reproduce it by transferring the files to the computer.

They will be in the directory */data/user_storage*. It will be a file with an extension .xlsx and have the name provided in the input variable file

Command

Transferring files from OT to computer (Linux, macOS)

Transferring files from OT to computer

```
scp -i [path_ot_key]
root@[IP_robot]:/data/user_storage/[name_map].xlsx
[final_path_computer]
```

Expected result

The map(s) contains a table for each final plate in the run

Each table will have the name given in the Combinations sheet of the input variable which has to be unique so there is no confusion on the placement of the different combinactions

Example

9 **We want to make 47 constructs that are all level 1 from 38 parts, some created in the lab, some from the golden standard database.**

Citation

Blázquez B, León DS, Torres-Bacete J, Gómez-Luengo Á, Kniewel R, Martínez I, Sordon S, Wilczak A, Salgado S, Huszcza E, Popłoński J, Prieto A, Nogales J
(2023)
. Golden Standard: a complete standard, portable, and interoperative MoClo tool for model and non-model proteobacteria..

https://doi.org/10.1093/nar/gkad758 LINK

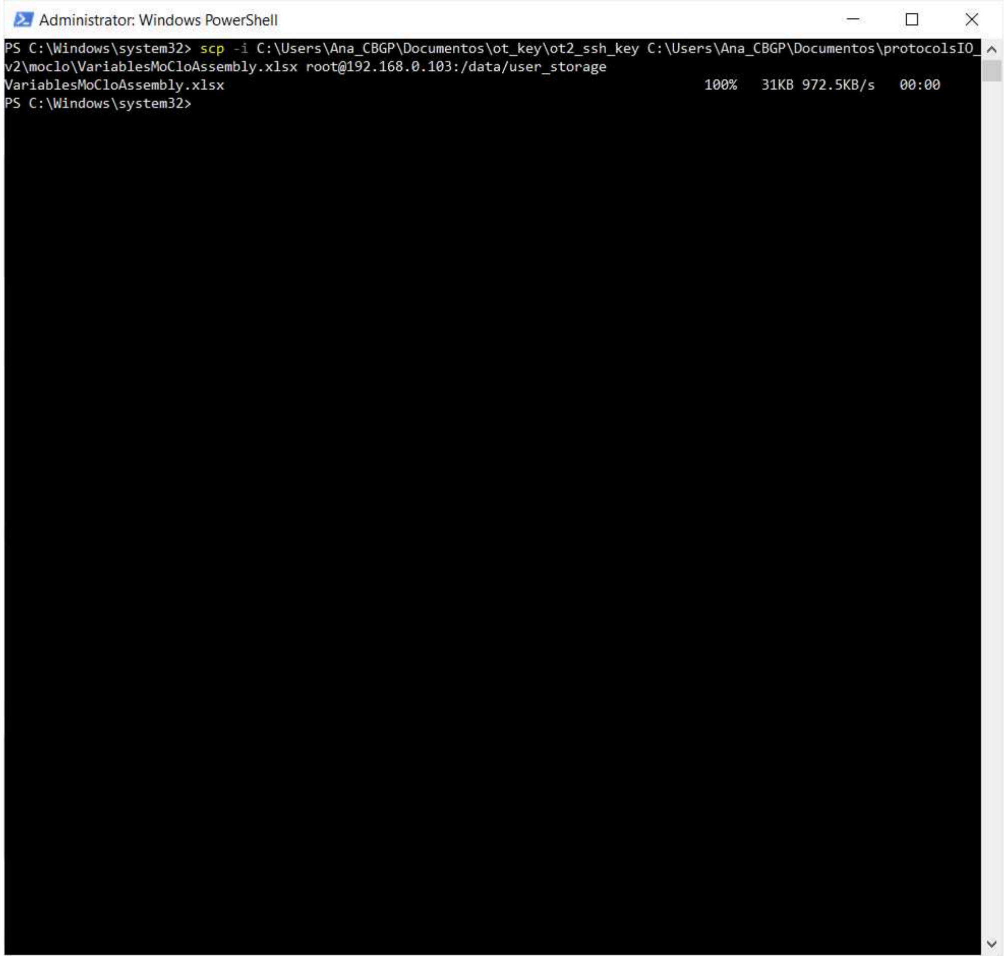
We are going to use the heater-shaker module to mix the assembly mix (enzyme, ligase, serum and buffer) and use a thermocycler module to perform the construct assembly with a provided temperature program.

9.1 Prepare variable file

Excel temple filled and saved with the name *VariablesMoCloAssembly.xlsx*

VariablesMoCloAssembly.xlsx

9.2 Export the variable file to *the /data/user_storage* folder in the robot



```
PS C:\Windows\system32> scp -i C:\Users\Ana_CBGP\Documentos\ot_key\ot2_ssh_key C:\Users\Ana_CBGP\Documentos\protocolsIO_
v2\moclo\VariablesMoCloAssembly.xlsx root@192.168.0.103:/data/user_storage
VariablesMoCloAssembly.xlsx                                                 100%   31KB 972.5KB/s   00:00
PS C:\Windows\system32>
```

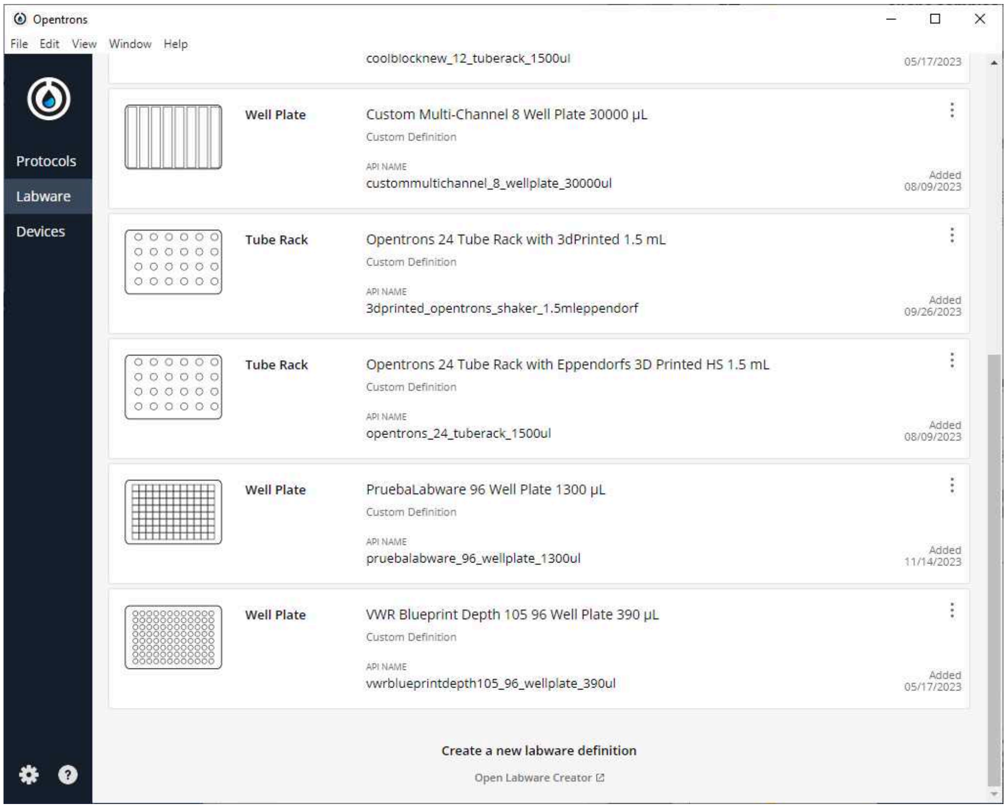
Command line window with scp commands to transfer the variables .xlsx from our computer to the OT-2

9.3 Upload custom labware to app

We are using a custom labware called *3dprinted_opentrons_shaker_1.5mleppendorf* that has been created with a 3D printer and its file with the labware creator that opentrons offers (**https://labware.opentrons.com/create/**)

FILE 3dprinted_opentrons_shaker_1.5mle... 3KB

Upload it to the opentrons app and make sure it is loaded in it



coolblocknew_12_tuberack_1500ul 05/17/2023

Well Plate — Custom Multi-Channel 8 Well Plate 30000 µL — Custom Definition — API NAME custommultichannel_8_wellplate_30000ul — Added 08/09/2023

Tube Rack — Opentrons 24 Tube Rack with 3dPrinted 1.5 mL — Custom Definition — API NAME 3dprinted_opentrons_shaker_1.5mleppendorf — Added 09/26/2023

Tube Rack — Opentrons 24 Tube Rack with Eppendorfs 3D Printed HS 1.5 mL — Custom Definition — API NAME opentrons_24_tuberack_1500ul — Added 08/09/2023

Well Plate — PruebaLabware 96 Well Plate 1300 µL — Custom Definition — API NAME pruebalabware_96_wellplate_1300ul — Added 11/14/2023

Well Plate — VWR Blueprint Depth 105 96 Well Plate 390 µL — Custom Definition — API NAME vwrblueprintdepth105_96_wellplate_390ul — Added 05/17/2023

List of custom labware recorded in the Opentrons App

9.4 Because we are using version 1.0.1 of the script in this example, we will **transfer the directory of the labware as well** (here we have attached a zip, but it is the folder that must be transferred, not the zip)

ZIP 3dprinted_opentrons_shaker_1.5mle...

Command

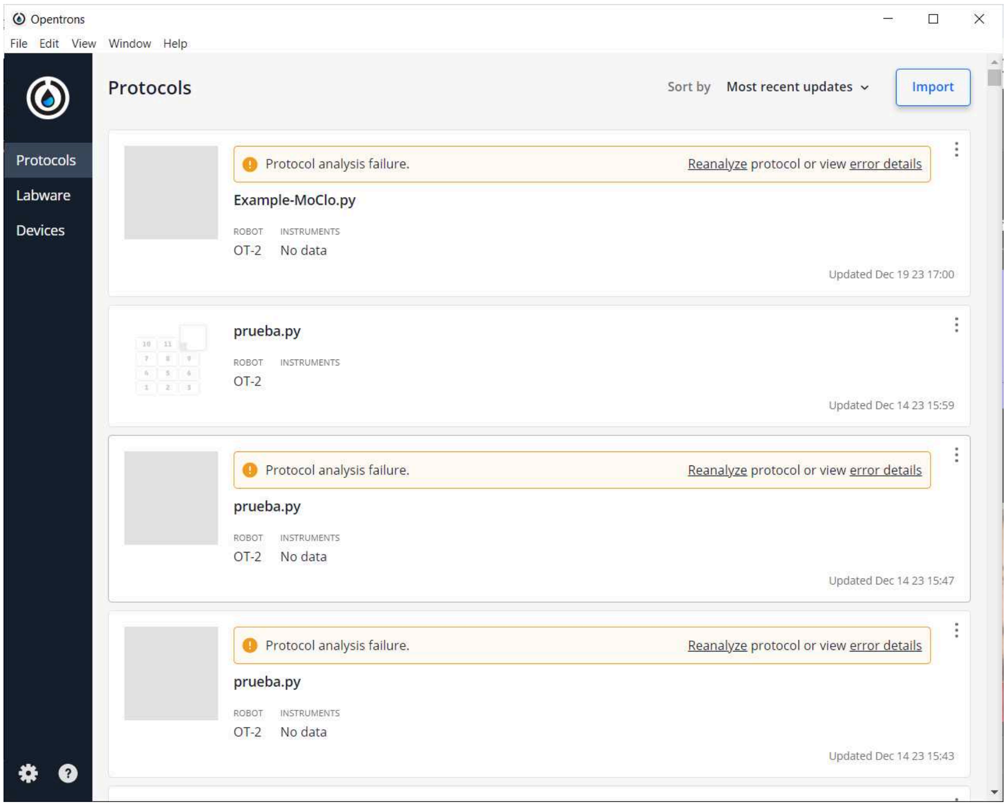
Transferring the used custom labware to OT (Linux)

Transferring the used custom labware to OT

```
scp -i [ot_key] -r 3dprinted_opentrons_shaker_1.5mleppendorf
root@[IP_OT]:/data/labware/v2/custom_definitions/custom_beta
```

9.5 **Import the script** that we have downloaded from the step go to step #5.1 (I named it *Example-MoClo.py*) to the OT-App

Example-MoClo.py



Protocol analysis failure. Reanalyze protocol or view error details

Example-MoClo.py

ROBOT OT-2 INSTRUMENTS No data

Updated Dec 19 23 17:00

prueba.py

ROBOT OT-2 INSTRUMENTS

Updated Dec 14 23 15:59

Protocol analysis failure. Reanalyze protocol or view error details

prueba.py

ROBOT OT-2 INSTRUMENTS No data

Updated Dec 14 23 15:47

Protocol analysis failure. Reanalyze protocol or view error details

prueba.py

ROBOT OT-2 INSTRUMENTS No data

Updated Dec 14 23 15:43

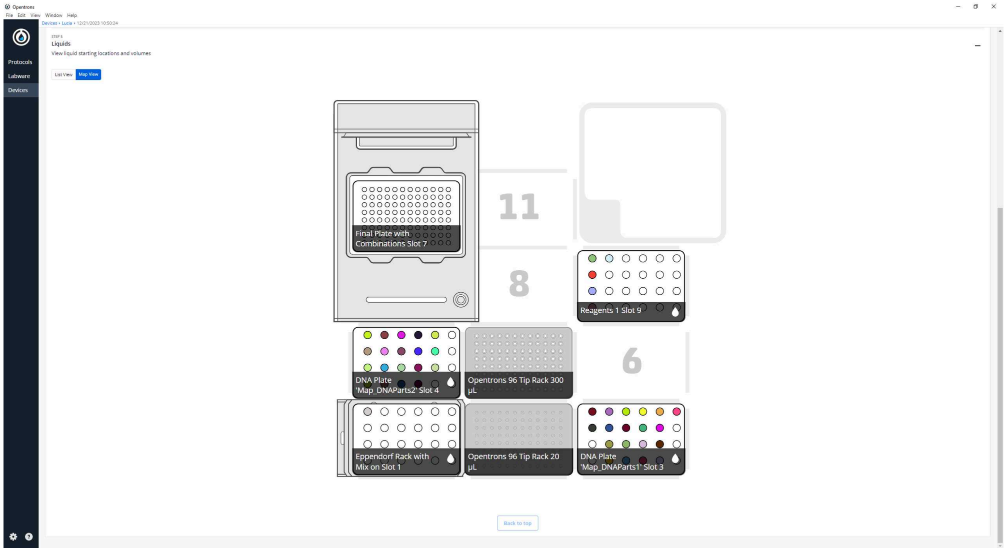
The result of importing the Python script into the OT-App

As we can see, we have an error, but that is programmed because the script is meant to work in the robot but not in your computer

9.6 **Run the protocol** in the robot that we have transferred the Excel file

Example-MoClo.py **→ Start setup → Select robot** in which we are going to run the protocol

If we do not have any errors, the output should look similar to the following pictures

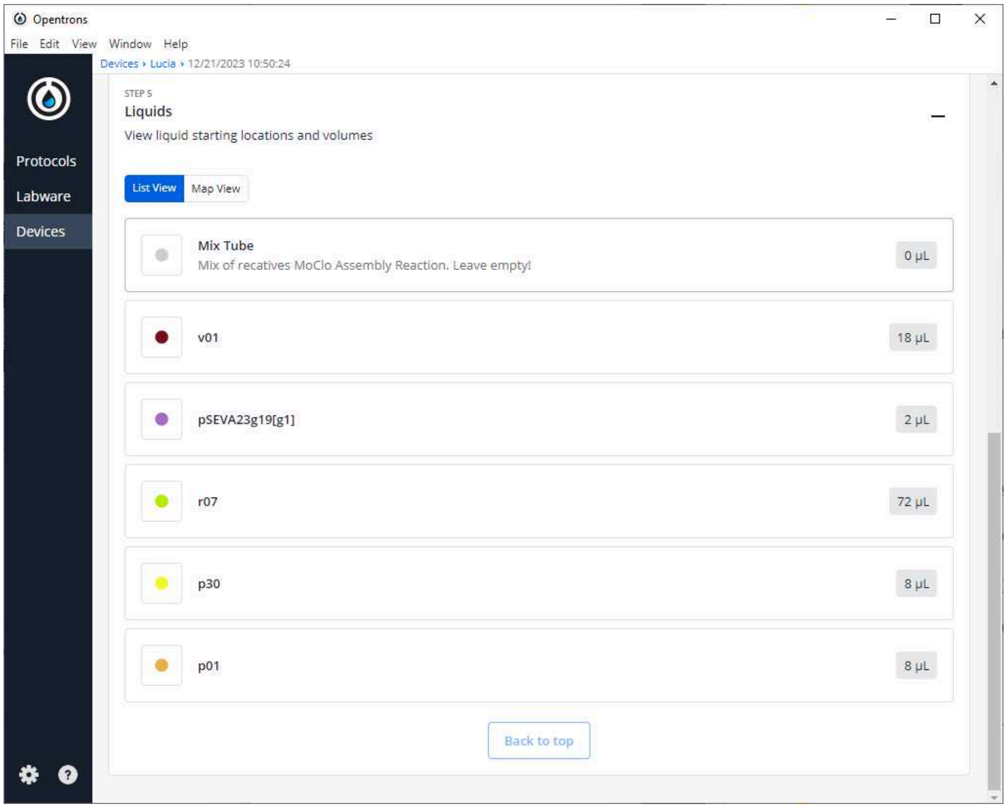


STEP 5

Liquids

View liquid starting locations and volumes

Labware and liquid set-up layout that corresponds to the run of the example variable input file



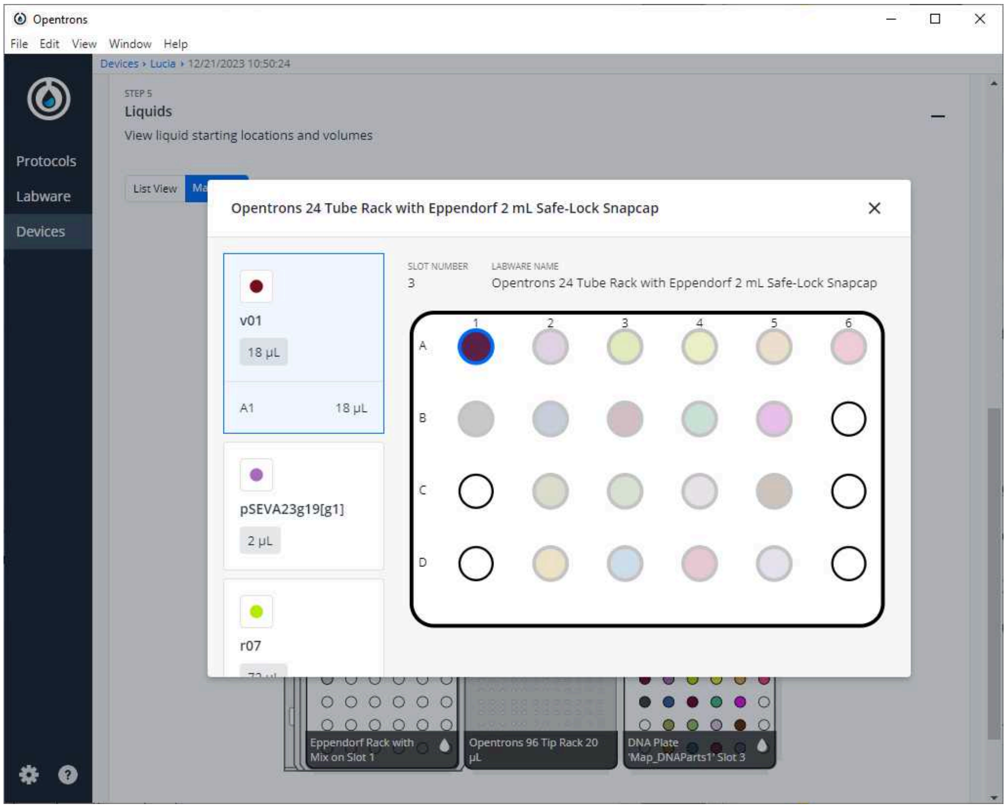
STEP 5

Liquids

View liquid starting locations and volumes

●	**Mix Tube** Mix of recatives MoClo Assembly Reaction. Leave empty!	0 µL
●	v01	18 µL
●	pSEVA23g19[g1]	2 µL
●	r07	72 µL
●	p30	8 µL
●	p01	8 µL

Volumes of the different reagents needed to perform the protocol that corresponds to the run of the example variable input file



STEP 5
Liquids
View liquid starting locations and volumes

Opentrons 24 Tube Rack with Eppendorf 2 mL Safe-Lock Snapcap

SLOT NUMBER: 3
LABWARE NAME: Opentrons 24 Tube Rack with Eppendorf 2 mL Safe-Lock Snapcap

v01 — 18 µL
A1 — 18 µL
pSEVA23g19[g1] — 2 µL
r07

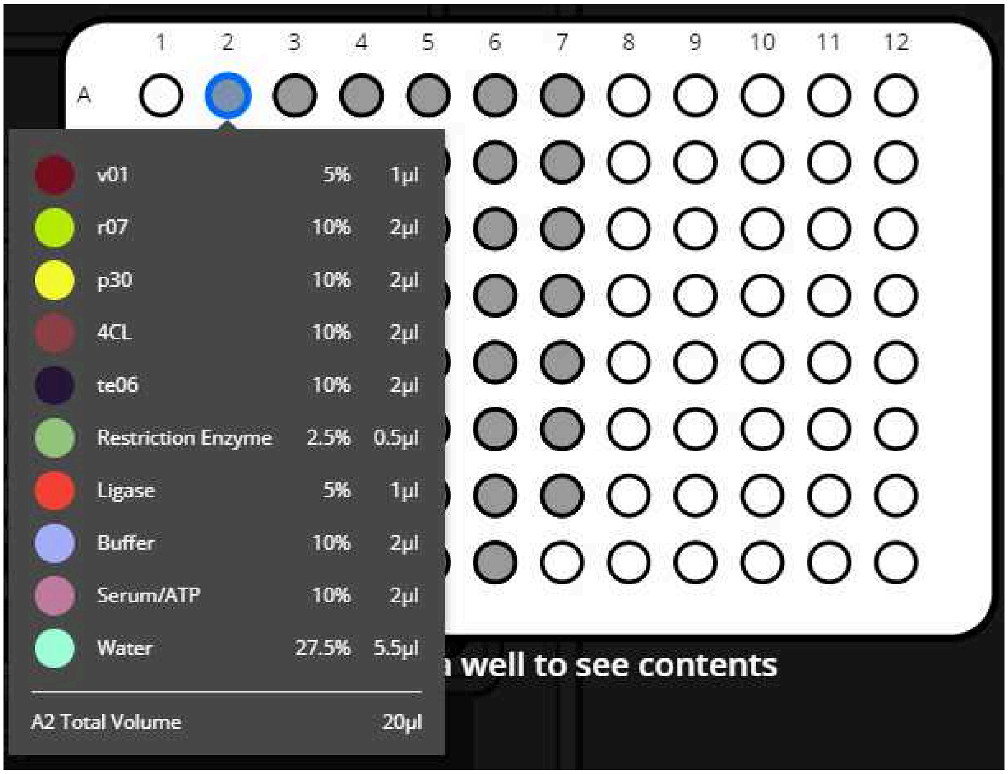
Positions and volumes of different DNA parts. The volumes are calculated but the positions are the ones set in the input file

9.7 **Clean the platform** of the robot that we are going to perform the protocol

9.8 **Prepare all reagents and labware** in the places as the App is showing taking into account the notes in step go to step #5.2 Notes

9.9 **Start run**

Expected result



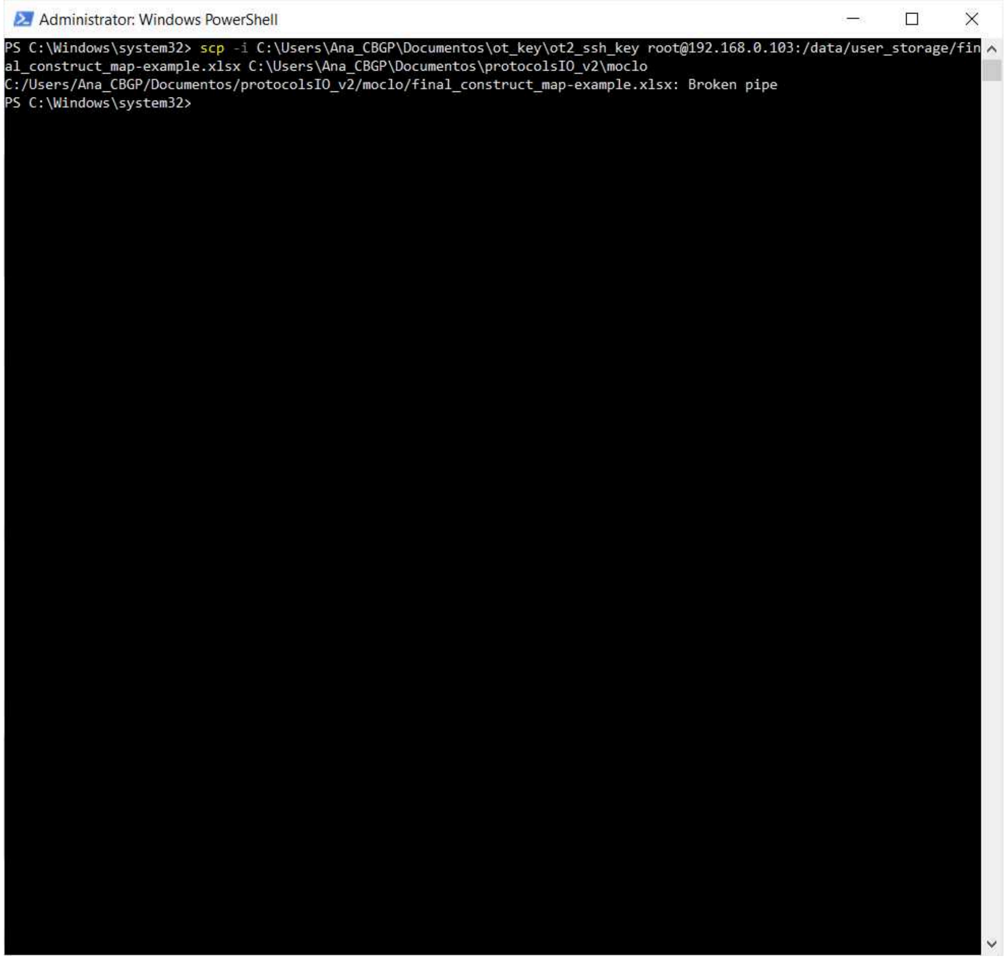
●	v01	5%	1µl
●	r07	10%	2µl
●	p30	10%	2µl
●	4CL	10%	2µl
●	te06	10%	2µl
●	Restriction Enzyme	2.5%	0.5µl
●	Ligase	5%	1µl
●	Buffer	10%	2µl
●	Serum/ATP	10%	2µl
●	Water	27.5%	5.5µl
	A2 Total Volume		20µl

Example of the content of a well, in this case A1 in the labware *Final Plate 1 with Combinations Slot 7*

Here, we will obtain the modular cloning mix and the diffrent DNA parts that are set in the input file. These positions are seen in the image by the grey wells, and we can see the info of the plate and the media in the plate on slot 7 in this case

9.10 **Retrieve labwares** from the OT

9.11 **Retrieve the final map(s) file from the robot** where we run the protocol. In this case, they will be called *final_construct_map-example.xlsx* (name that is stated in the variable file in the variable *Name File Final Constructs* located in the *GeneralVariablesSheet*)



```
PS C:\Windows\system32> scp -i C:\Users\Ana_CBGP\Documentos\ot_key\ot2_ssh_key root@192.168.0.103:/data/user_storage/fin
al_construct_map-example.xlsx C:\Users\Ana_CBGP\Documentos\protocolsIO_v2\moclo
C:/Users/Ana_CBGP/Documentos/protocolsIO_v2/moclo/final_construct_map-example.xlsx: Broken pipe
PS C:\Windows\system32>
```

Command line window with the transfer command of the final file with the map(s) from the OT to our computer

final_construct_map-example.xlsx

Protocol references

The Laboratory Automation Protocol (LAP) Format and Repository: A Platform for Enhancing Workflow Efficiency in Synthetic Biology (*ACS Synth. Biol.*) https://doi.org/10.1021/acssynbio.3c00397

Golden Standard: a complete standard, portable, and interoperative MoClo tool for model and non-model proteobacteria *(Nucleic Acids Research)* https://doi.org/10.1093/nar/gkad758

Citations

Step 9

Blázquez B, León DS, Torres-Bacete J, Gómez-Luengo Á, Kniewel R, Martínez I, Sordon S, Wilczak A, Salgado S, Huszcza E, Popłoński J, Prieto A, Nogales J. Golden Standard: a complete standard, portable, and interoperative MoClo tool for model and non-model proteobacteria.
https://doi.org/10.1093/nar/gkad758