



自動分注ロボット OT-2

プロトコル作成について ~基礎編~

OT-2使用の流れ



①プロトコル作成 = どういった動作をさせるか決める。



(②チップ, ピペットのキャリブレーション)



③プロトコルのアップロード



④ポジションチェック(チップラック, プレート, チューブラック)



⑤プログラム実行！

OT-2使用の流れ

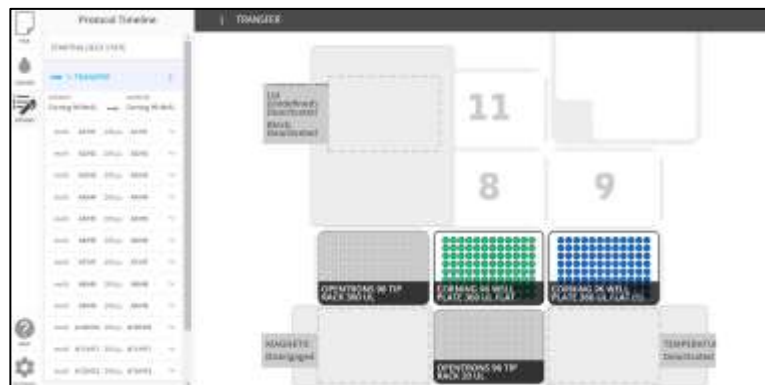


- 1: Python APIを用いた作成
 - ⇒自由度が高いプロトコル作成方法
 - ⇒ある程度Pythonの知識が必要

```
1 from opentrons import load_module, load_station, robot
2
3 pipette = load_module('pipette', '1')
4 plate = load_module('96-flat', '2')
5 plate2 = load_module('96-flat', '3')
6 plate3 = load_module('96-flat', '4')
7 plate4 = load_module('96-flat', '5')
8 plate5 = load_module('96-flat', '6')
9 plate6 = load_module('96-flat', '7')
10 plate7 = load_module('96-flat', '8')
11 plate8 = load_module('96-flat', '9')
12 plate9 = load_module('96-flat', '10')
13
14 pip = load_module('p10-single', '10', pip_name='pipette')
15
16 pip.transfer(10, plate1.wells[100], plate1.wells[101])
17 pip.transfer(10, plate1.wells[102], plate1.wells[103])
18 pip.transfer(10, plate1.wells[104], plate1.wells[105])
19 pip.transfer(10, plate1.wells[106], plate1.wells[107])
20
```

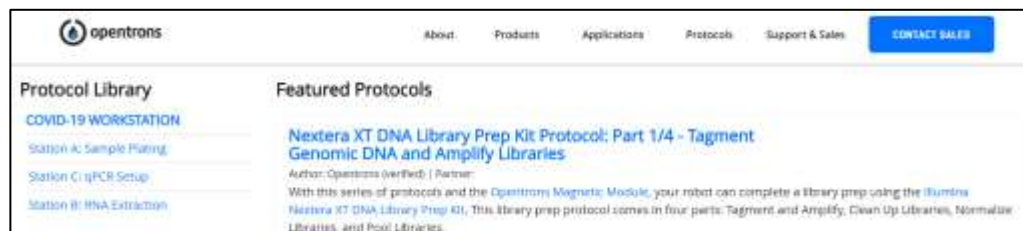
2: Protocol Designer

- ⇒モジュールの使用を含めて自分好みのプロトコルを作成できる
- ⇒ウェブベースで簡単に利用でき、視覚的にもわかりやすい



3: Opentrons webからダウンロード

- ⇒自分の目的に当てはまるものがあればすぐに使える



プロトコル作成1: Python APIを用いた作成



- テキストベースで自由度が高いプログラムが作成できます
- × 作成のルールやコマンドを覚える(調べる)必要があります

```
1 h_asp = 1
2 h_dis = 1
3
4 from opentrons import protocol_api
5
6 metadata = {'apiLevel': '2.11'}
7
8 def run(protocol: protocol_api.ProtocolContext):
9
10     tiprack = protocol.load_labware('opentrons_96_tiprack_300ul', '6')
11     tiprack2 = protocol.load_labware('opentrons_96_tiprack_20ul', '9')
12     dis_plate = protocol.load_labware('biorad_96_wellplate_200ul_pcr', '3')
13
14     rack1 = protocol.load_labware('biorad_96_wellplate_200ul_pcr', '1')
15     rack2 = protocol.load_labware('biorad_96_wellplate_200ul_pcr', '2')
16
17     p300 = protocol.load_instrument('p300_single_gen2', 'left', tip_racks=[tiprack])
18
19     p20 = protocol.load_instrument('p20_single_gen2', 'right', tip_racks=[tiprack2])
20
21     p20.transfer(10, rack1['A1'].bottom(h_asp), dis_plate['A1'].bottom(h_dis))
22     p300.transfer(30, rack1['B1'].bottom(h_asp), dis_plate['B1'].bottom(h_dis))
```

プロトコル作成1: Python APIを用いた作成



①ソフトウェアの準備

Sublime Text

下記URLより、使用するPCのOSにあったバージョンをダウンロードしてください
Pythonプロトコルを作成するために使用される一般的なコードエディターです

<https://www.sublimetext.com/3>

Sublime Textを起動すると下記の画面が表示されます
※特にこのソフトでなくても問題ございません

```
1 h_asp = 1
2 h_dis = 1
3
4 from opentrons import protocol_api
5
6 metadata = {'apiLevel': '2.11'}
7
8 def run(protocol: protocol_api.ProtocolContext):
9
10     tiprack = protocol.load_labware('opentrons_96_tiprack_300ul', '6')
11     tiprack2 = protocol.load_labware('opentrons_96_tiprack_300ul', '9')
12     dis_plate = protocol.load_labware('biorad_96_wellplate_200ul_pcr', '3')
13
14     rack1 = protocol.load_labware('biorad_96_wellplate_200ul_pcr', '1')
15     rack2 = protocol.load_labware('biorad_96_wellplate_200ul_pcr', '2')
16
17     p300 = protocol.load_instrument('p300_single_gen2', 'left', tip_racks=[tiprack])
18
19     p20 = protocol.load_instrument('p20_single_gen2', 'right', tip_racks=[tiprack2])
20
21     p20.transfer(10, rack1['A1'].bottom(h_asp), dis_plate['A1'].bottom(h_dis))
22     p300.transfer(30, rack1['B1'].bottom(h_asp), dis_plate['B1'].bottom(h_dis))
23     p300.transfer(100, rack1['C1'].bottom(h_asp), dis_plate['C1'].bottom(h_dis))
24     p300.transfer(150, rack1['D1'].bottom(h_asp), dis_plate['D1'].bottom(h_dis))
25     p300.transfer(190, rack2['A1'].bottom(h_asp), dis_plate['A1'].bottom(h_dis))
26     p300.transfer(170, rack2['A1'].bottom(h_asp), dis_plate['B1'].bottom(h_dis))
27     p300.transfer(180, rack2['A1'].bottom(h_asp), dis_plate['C1'].bottom(h_dis))
28     p300.transfer(50, rack2['A1'].bottom(h_asp), dis_plate['D1'].bottom(h_dis))
29
```

- 赤枠の内部にプロトコルをタイプします
- 保存の際は、左上に表示される
File > Save as で保存場所とファイル名を決定します
- 保存時の拡張子は【.py】を選択します
- Sublime Text以外のコードエディターを使用した場合
も同様に【.py】で保存します

プロトコル作成1: Python APIを用いた作成



②プロトコルの記述方法(基本)

```
from opentrons import protocol_api, types
```

```
metadata = {'apiLevel': '2.11'}
```

```
def run(protocol):
```

```
    plate = protocol.load_labware('corning_96_wellplate_360ul_flat', '2')
```

```
    tiprack = protocol.load_labware('opentrons_96_tiprack_300ul', '1')
```

```
    left_pipette = protocol.load_instrument('p300_single_gen2', 'left', tip_racks=[tiprack])
```

```
    left_pipette.pick_up_tip()
```

```
    left_pipette.aspirate(100, plate['A1'])
```

```
    left_pipette.dispense(100, plate['B2'])
```

```
    left_pipette.drop_tip()
```

プロトコル作成1: Python APIを用いた作成



②プロトコルの記述方法(基本)

```
from opentrons import protocol_api, types
```

```
metadata = {'apiLevel': '2.11'}
```

```
def run(protocol):
```

```
    タブ plate = protocol.load_labware('corning_96_wellplate_360ul_flat', '2')
```

```
    タブ tiprack = protocol.load_labware('opentrons_96_tiprack_300ul', '1')
```

```
    タブ left_pipette = protocol.load_instrument('p300_single_gen2', 'left', tip_racks=[tiprack])
```

```
    タブ left_pipette.pick_up_tip()
```

```
    タブ left_pipette.aspirate(100, plate['A1'])
```

```
    タブ left_pipette.dispense(100, plate['B2'])
```

```
    タブ left_pipette.drop_tip()
```

ルール: 1つのグループはインデント(字下げ)を行う(Pythonのルール)

プロトコル作成1: Python APIを用いた作成



②プロトコルの記述方法(基本)

```
from opentrons import protocol_api, types
```

定型文

```
metadata = {'apiLevel': '2.11'}
```

バージョン管理

```
def run(protocol):
```

定型文

```
    タブ plate = protocol.load_labware('corning_96_wellplate_360ul_flat', '2')
```

```
    タブ tiprack = protocol.load_labware('opentrons_96_tiprack_300ul', '1')
```

使用するラボウェアの
定義

```
    タブ left_pipette = protocol.load_instrument('p300_single_gen2', 'left', tip_racks=[tiprack])
```

ピペットの定義

```
    タブ left_pipette.pick_up_tip()
```

```
    タブ left_pipette.aspirate(100, plate['A1'])
```

```
    タブ left_pipette.dispense(100, plate['B2'])
```

```
    タブ left_pipette.drop_tip()
```

分注動作の定義

基本形は使用するラボウェア, ピペットを定義し, 分注動作を記述していきます

プロトコル作成1: Python APIを用いた作成



③バージョン管理について

```
metadata = {'apiLevel': '2.11'}
```

apiLevel (API version)が重要です

API versionによって対応するOT-2ソフトウェアが異なります

また, API versionによって実装される機能が異なることがあります

API Version	Introduced In OT-2 Software
1.0	3.0.0
2.0	3.14.0
2.1	3.15.2
2.2	3.16.0
...	...
2.11	4.4.0

できる限り最新バージョンで記述いただく事をお勧めします

最新バージョンや各バージョンの情報は

<https://docs.opentrons.com/v2/versioning.html> をご覧ください

プロトコル作成1: Python APIを用いた作成



④使用するラボウェア(チップラック, チューブ, プレート)の定義 その1

```
タブ plate = protocol.load_labware('corning_96_wellplate_360ul_flat', '2')
```

```
タブ tiprack = protocol.load_labware('opentrons_96_tiprack_300ul', '1')
```

```
タブ 任意の名前 = protocol.load_labware ('Opentrons指定の名前', '設置場所')
```

好きな名前をつけます
プログラム内ではこの名前を使用します

Opentrons社のウェブから適切なものを選びます
→次ページへ

1~11の好きな場所を選びます



ルール

1. 装着するピペットに適合するチップラックを必ず用意する。
2. 設置場所は重複できない。
3. 単語中にスペースは使用しない。(Python)
4. 任意の名前の1文字目に数字を用いない。(Python)

プロトコル作成1: Python APIを用いた作成



④使用するラボウェア(チップラック, チューブ, プレート)の定義 その2

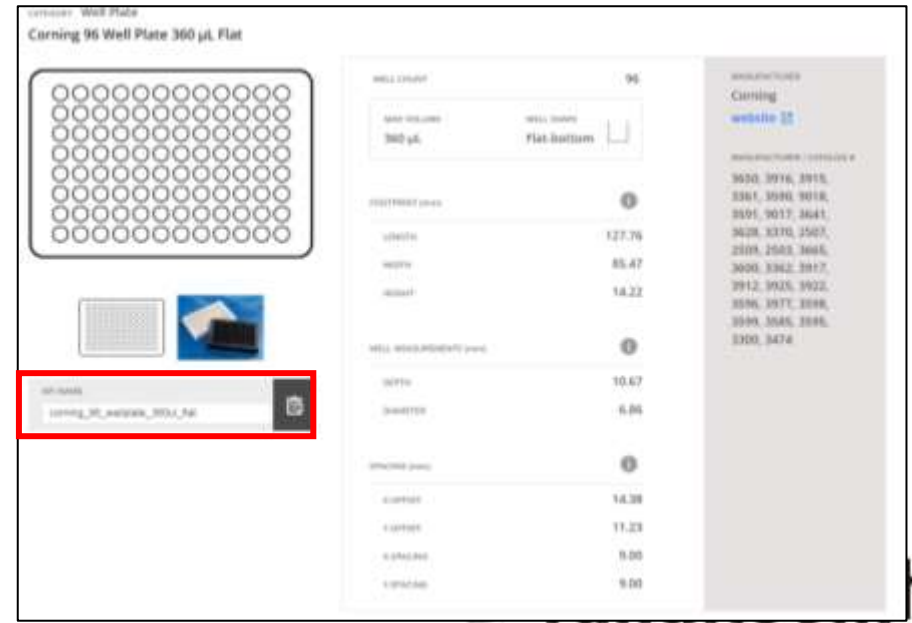
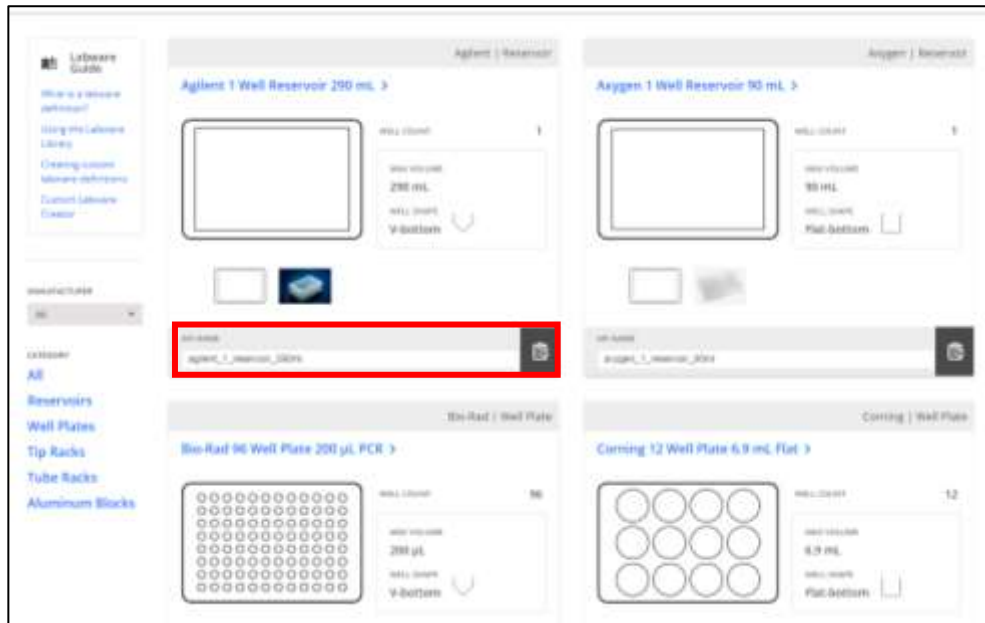
任意の名前 = protocol.load_labware ('Opentrons指定の名前', '設置場所')

Opentrons指定の名称はOpentrons WEBに設置されているLabware Libraryを参照します
<https://labware.opentrons.com/>

各ラボウェアに記載されている「API NAME」を記述します

例: corning_96_wellplate_360ul_flat

opentrons_96_aluminumblock_generic_pcr_strip_200ul



プロトコル作成1: Python APIを用いた作成



⑤使用するピペットの定義

```
left_pipette = protocol.load_instrument('p300_single_gen2', 'left', tip_racks=[tiprack])
```

任意の名前

```
= protocol.load_instrument('ピペットの種類', '左右', tip_racks=[チップラック名])
```

装着するピペットの
種類を指定します
下記表を参照

ピペットを装着するマウントを指定します
【left】か【right】で指定します

チップラックの定義で
指定した任意の名前を記載します

Pipette Type	Model Name
P20 Single GEN2 (1 - 20 μ L)	'p20_single_gen2'
P300 Single GEN2 (20 - 300 μ L)	'p300_single_gen2'
P1000 Single GEN2 (100 - 1000 μ L)	'p1000_single_gen2'
P20 Multi GEN2 (1 - 20 μ L)	'p20_multi_gen2'
P300 Multi GEN2 (20 - 300 μ L)	'p300_multi_gen2'

プロトコル作成1: Python APIを用いた作成



⑤分注動作の定義 その1

```
left_pipette.pick_up_tip()
left_pipette.aspirate(100, plate['A1'])
left_pipette.dispense(100, plate['B2'])
left_pipette.drop_tip()
```

ピペット名.動作('容量', '場所')

使用するピペットを指定します
ピペットの定義で指定した名前を使用します

動作を指定します

動作する場所

動作	記述法
チップの装着	pick_up_tip
チップの取り外し	drop_tip
吸引	aspirate
排出	dispense



⑤分注動作の定義 その2

`ピペット名.動作('容量', '場所')`

`left_pipette.pick_up_tip()` チップの装着

`left_pipette.aspirate(100, plate['A1'])` left_pipetteでプレートのA1から100ulを吸引

`left_pipette.dispense(100, plate['B2'])` left_pipetteでプレートのB2へ100ulを排出

`left_pipette.drop_tip()` チップの取り外し

`left_pipette.transfer(100, plate['A1'], plate['B2'])`

P10でプレートのA1ウェルから10ulを吸引し、プレートのB2ウェルへ排出

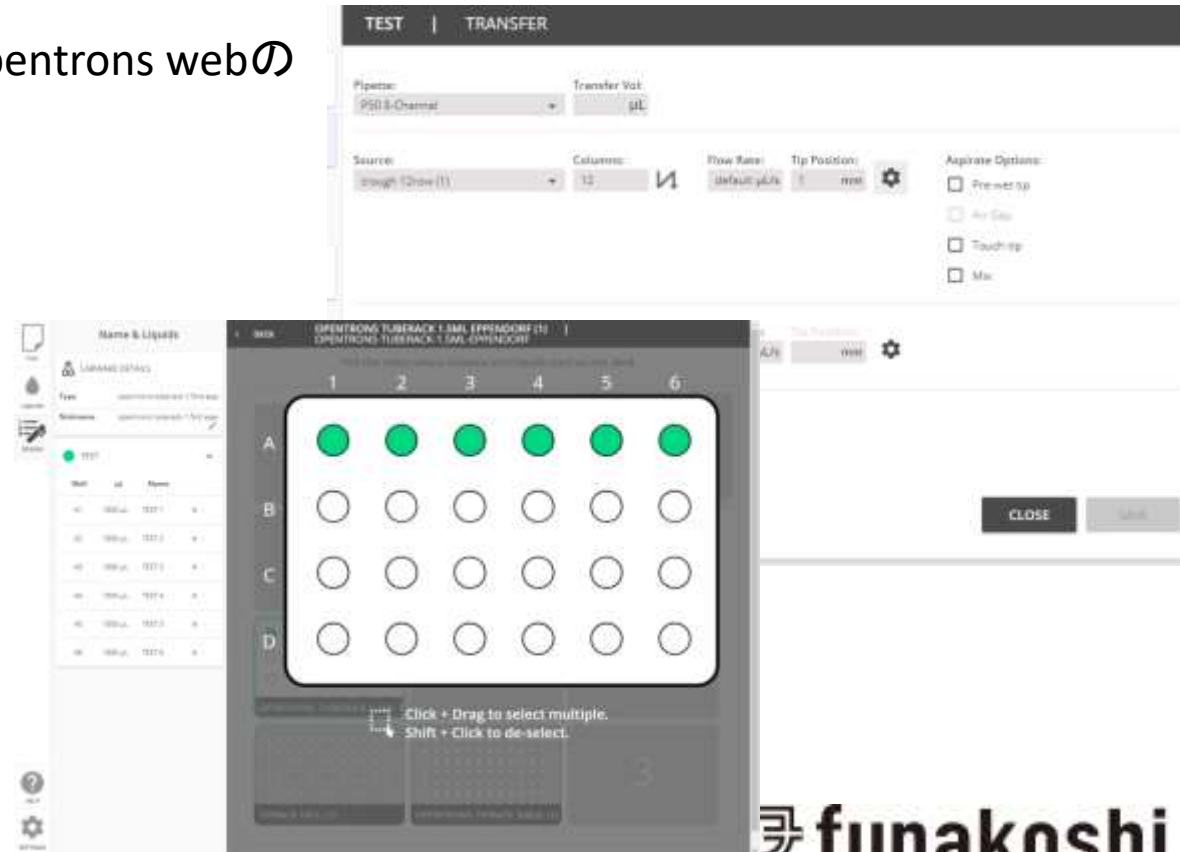
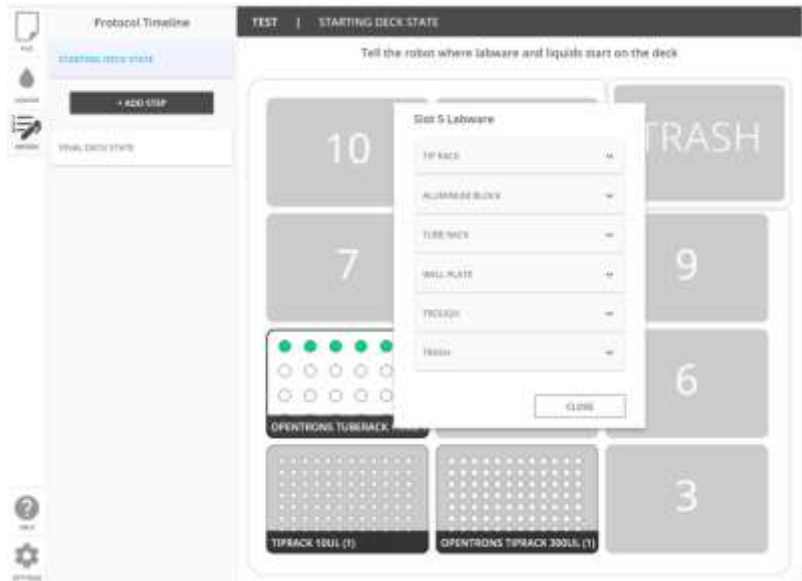
その他多種の動作コマンドが用意されています
最新情報、詳細情報は
<https://docs.opentrons.com/v2/> をご確認ください

プロトコル作成2: Protocol Designerを用いた作成



- 直感的な操作でプログラムを作成できるwebアプリケーションです
- ✕ PCごとにユーザー登録が必要です
Python API詳細設定ができない場合があります
編集はProtocol Designer内のみで可能です(テキストベースには変換できません)

Protocol DesignerへはアクセスはOpentrons webの
トップページからアクセスできます



プロトコル作成3: Opentrons webからダウンロード



○ ダウンロードするだけで使用できます

× 目的に合ったプロトコルが無い可能性があります
微調整を行う場合、ダウンロードしたプロトコルをPython APIでの編集が必要

Protocol Library > Nucleic Acid Extraction & Purification > Viral RNA > Beckman Coulter RNAdvance Viral RNA Isolation

COVID WORKSTATION:

- qPCR Setup
- Sample Plating
- RNA Extraction

FEATURED:

- PCR Prep
- NGS Library Prep: Illumina Nextera XT
- NGS Library Prep: Swift 25 Turbo
- Normalization
- NGS Cleanup and Size Selection with Omega Bio-tek Mag-Bind® TotalPure
- Cherry-picking
- Nucleic Acid Purification with Magnetic Beads (Universal)

Beckman Coulter RNAdvance Viral RNA Isolation

Author: Opentrons | Partner Lab: Beckman Coulter Life Sciences

This protocol performs viral RNA isolation on up to 96 samples using the Beckman Coulter RNAdvance Viral RNA Isolation kit and workflow.

The protocol begins at the stage of adding binding beads to lysed samples in a NEST 96-deepwell plate. For reagent layout in the 2 12-channel reservoir below.

For sample traceability and consistency, samples are mapped directly from the NEST 96-deepwell plate to the elution PCR plate (temperature module, slot 1). Samples are then transferred to elution PCR plate A1, extraction plate well B1 to elution plate.

For more information on the Beckman Coulter RNAdvance Viral RNA Isolation kit, see the Application Note.

Modules: Temperature Module (GEN2) | Magnetic Module (GEN2)

Labware:

- NEST 12 Well Reservoir 15 mL 2x
- NEST 1 Well Reservoir 195 mL
- NEST 96 Well Plate 100 µL PCR Full Skirt
- NEST 96 Deepwell Plate 2mL
- Opentrons 96 Filter Tip Rack 200 µL, up to 10x depending on throughput

Reagent Reservoirs:

- REAGENT RESERVOIR 1:** 12 channels. Channels 1-2: Bind BBD + Isopropanol (purple); Channels 3-4: Wash WBE (green); Channels 5-6: 70% Ethanol (blue); Channels 7-12: Empty.
- REAGENT RESERVOIR 2:** 12 channels. Channels 1-2: 70% Ethanol (pink); Channels 3-11: Empty; Channel 12: Nuclease-free water (red).

Reservoir 2: slot 3

Volumes per reservoir channel: (for 96-sample run, not including dead volume):

- 10mL of 100% isopropanol + 250µL of bead BBD
- 10mL of Wash WBE
- 10mL of 70% Ethanol
- 4mL of nuclease-free water

OT-2

Download Protocol:

number of samples + controls (1-96):

starting volume (100-400µl):

elution volume (50-100µl):