opentrons

You are currently viewing the documentation for the Opentrons OT-1. To view documentation for the OT-2, click here.

Transfer Shortcuts

The Transfer command is a nice way to wrap up the most common liquid-handling actions we take. Instead of having to write `loop` and `if` statements, we can simply use the `transfer()` command, making Python protocol both easier to write and read!

Transfer

Most of time, a protocol is really just looping over some wells, aspirating, and then dispensing. Even though they are simple in nature, these loops take up a lot of space. The `pipette.transfer()` command takes care of those common loops. It will combine aspirates and dispenses automatically, making your protocol easier to read and edit.

```
'''
Examples in this section expect the following
'''
from opentrons import containers, instruments

plate = containers.load('96-flat', 'B1')

tiprack = containers.load('tiprack-200ul', 'A1')
trash = containers.load('point', 'D2')

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

Basic

The example below will transfer 100 uL from well `'A1'` to well `'B1'`, automatically picking up a new tip and then dropping it when finished.

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

Transfer commands will automatically create entire series of `aspirate()`, `dispense()`, and other `Pipette` commands. We can print out all commands to see what it did in the previous example:

```
for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B1>
```

Volumes larger than the pipette's `max_volume` will automatically divide into smaller transfers.

```
pipette.transfer(700, plate.wells('A2'), plate.wells('B2'))

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 200.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 200.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Aspirating 200.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 200.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Aspirating 150.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 150.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Aspirating 150.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 150.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

Multiple Wells

Transfer commands are most useful when moving liquid between multiple wells.

```
pipette.transfer(100, plate.cols('A'), plate.cols('B'))

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A3>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B3>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A4>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B4>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A5>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B5>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A6>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B6>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A7>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B7>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A8>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B8>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A9>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B9>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A10>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B10>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A11>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B11>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A12>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B12>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

One to Many

You can transfer from a single source to multiple destinations, and the other way around (many sources to one destination).

```
pipette.transfer(100, plate.wells('A1'), plate.rows('2'))
```

opentrons

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well C2>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well D2>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well E2>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well F2>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well G2>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well H2>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

Few to Many

What happens if, for example, you tell your pipette to transfer from 4 source wells to 2 destination wells? The transfer command will attempt to divide the wells evenly, or raise an error if the number of wells aren't divisible.

```
pipette.transfer(
    100,
    plate.wells('A1', 'A2', 'A3', 'A4'),
    plate.wells('B1', 'B2'))

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A3>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A4>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

List of Volumes

Instead of applying a single volume amount to all source/destination wells, you can instead pass a list of volumes.

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

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 20.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 20.0 at <Deck><Slot B1><Container 96-flat><Well B1>
```

opentrons

Volume Gradient

Create a linear gradient between a start and ending volume (uL). The start and ending volumes must be the first and second elements of a tuple.

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

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Aspirating 90.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 90.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Aspirating 80.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 80.0 at <Deck><Slot B1><Container 96-flat><Well C2>
Aspirating 70.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 70.0 at <Deck><Slot B1><Container 96-flat><Well D2>
Aspirating 60.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 60.0 at <Deck><Slot B1><Container 96-flat><Well E2>
Aspirating 50.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 50.0 at <Deck><Slot B1><Container 96-flat><Well F2>
Aspirating 40.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 40.0 at <Deck><Slot B1><Container 96-flat><Well G2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well H2>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

Distribute and Consolidate

Save time and tips with the `distribute()` and `consolidate()` commands. These are nearly identical to `transfer()`, except that they will combine multiple transfer's into a single tip.

```
'''
Examples in this section expect the following
'''
from opentrons import containers, instruments

plate = containers.load('96-flat', 'B1')

tiprack = containers.load('tiprack-200ul', 'A1')
trash = containers.load('point', 'D2')

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

Consolidate

Volumes going to the same destination well are combined within the same tip, so that multiple aspirates can be combined to a single dispense.

opentrons

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well C2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well D2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well E2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well F2>
Dispensing 180.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well G2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well H2>
Dispensing 60.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

If there are multiple destination wells, the pipette will never combine their volumes into the same tip.

```
pipette.consolidate(30, plate.rows('2'), plate.wells('A1', 'A2'))

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well C2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well D2>
Dispensing 120.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well E2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well F2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well G2>
Aspirating 30.0 at <Deck><Slot B1><Container 96-flat><Well H2>
Dispensing 120.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

Distribute

Volumes from the same source well are combined within the same tip, so that one aspirate can provide for multiple dispenses.

```
pipette.distribute(55, plate.wells('A1'), plate.rows('2'))

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 165.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 55.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 55.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Dispensing 55.0 at <Deck><Slot B1><Container 96-flat><Well C2>
Aspirating 165.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 55.0 at <Deck><Slot B1><Container 96-flat><Well D2>
Dispensing 55.0 at <Deck><Slot B1><Container 96-flat><Well E2>
Dispensing 55.0 at <Deck><Slot B1><Container 96-flat><Well F2>
Aspirating 110.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 55.0 at <Deck><Slot B1><Container 96-flat><Well G2>
Dispensing 55.0 at <Deck><Slot B1><Container 96-flat><Well H2>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

If there are multiple source wells, the pipette will never combine their volumes into the same tip.

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 120.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well C2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well D2>
Aspirating 120.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well E2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well F2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well G2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well H2>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

Disposal Volume

When dispensing multiple times from the same tip, it is recommended to aspirate an extra amount of liquid to be disposed of after distributing. This added `disposal_vol` can be set as an optional argument.

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

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 130.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well C2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well D2>
Blowing out at <Deck><Slot D2><Container point><Well A1>
Aspirating 130.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well E2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well F2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well G2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well H2>
Blowing out at <Deck><Slot D2><Container point><Well A1>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

Note:

If you do not specify a `disposal_vol`, the pipette will by default use a `disposal_vol` equal to it's `min_volume`. This tutorial has not given the pipette any `min_volume`, so below is an example of allowing the pipette's `min_volume` to be used as a default for `disposal_vol`.

```
pipette.min_volume = 20  # `min_volume` is used as default to `disposal_vol`

pipette.distribute(
    30,
    plate.wells('A1', 'A2'),
    plate.rows('2'))

for c in robot.commands():
    print(c)
```

```
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well C2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well D2>
Blowing out at <Deck><Slot D2><Container point><Well A1>
Aspirating 140.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well E2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well F2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well G2>
Dispensing 30.0 at <Deck><Slot B1><Container 96-flat><Well H2>
Blowing out at <Deck><Slot D2><Container point><Well A1>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

Transfer Options

There are other options for customizing your transfer command:

```
'''
Examples in this section expect the following
'''
from opentrons import containers, instruments

plate = containers.load('96-flat', 'B1')

tiprack = containers.load('tiprack-200ul', 'A1')
trash = containers.load('point', 'D2')

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

Always Get a New Tip

Transfer commands will by default use the same one tip for each well, then finally drop it in the trash once finished.

The pipette can optionally get a new tip at the beginning of each aspirate, to help avoid cross contamination.

```
pipette.transfer(
    100,
    plate.wells('A1', 'A2', 'A3'),
    plate.wells('B1', 'B2', 'B3'),
    new_tip='always')    # always pick up a new tip

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B1>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well B1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well C1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A3>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B3>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

ignore picking up or dropping tips.

```
pipette.transfer(
    100,
    plate.wells('A1', 'A2', 'A3'),
    plate.wells('B1', 'B2', 'B3'),
    new_tip='never')    # never pick up or drop a tip

for c in robot.commands():
    print(c)
```

will print out...

```
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B2>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A3>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B3>
```

Trash or Return Tip

By default, the transfer command will drop the pipette's tips in the trash container. However, if you wish to instead return the tip to it's tip rack, you can set `trash=False`.

```
pipette.transfer(
    100,
    plate.wells('A1'),
    plate.wells('B1'),
    trash=False)       # do not trash tip

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well B1>
Returning tip
Drop_tip at <Deck><Slot A1><Container tiprack-200ul><Well A1>
```

Touch Tip

A touch-tip can be performed after every aspirate and dispense by setting `touch_tip=True`.

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

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Touching tip
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Touching tip
```

A blow-out can be performed after every dispense that leaves the tip empty by setting `blow_out=True`.

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

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Blowing out
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

Mix Before/After

A mix can be performed before every aspirate by setting `mix_before=`. The value of `mix_before=` must be a tuple, the 1st value is the number of repetitions, the 2nd value is the amount of liquid to mix.

```
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

for c in robot.commands():
    print(c)
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Mixing 2 times with a volume of 50ul
Aspirating 50 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 50
Aspirating 50
Dispensing 50
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Mixing 3 times with a volume of 75ul
Aspirating 75 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 75.0
Aspirating 75
Dispensing 75.0
Aspirating 75
Dispensing 75.0
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```

Air Gap

An air gap can be performed after every aspirate by setting `air_gap=int`, where the value is the volume of air in microliters to aspirate after aspirating the liquid.

```
pipette.transfer(
    100,
    plate.wells('A1'),
    plate.wells('A2'),
```

will print out...

```
Picking up tip from <Deck><Slot A1><Container tiprack-200ul><Well A1>
Aspirating 100.0 at <Deck><Slot B1><Container 96-flat><Well A1>
Air gap
Aspirating 20
Dispensing 20 at <Deck><Slot B1><Container 96-flat><Well A2>
Dispensing 100.0 at <Deck><Slot B1><Container 96-flat><Well A2>
Drop_tip at <Deck><Slot D2><Container point><Well A1>
```