

How to write a PR-PR script

PR-PR is a high-level language that enables Biologists to more quickly design experiments that utilize liquid-handling robotics.

General guidelines for PR-PR scripts:

- All lines beginning with the pound character ‘#’ are ignored by PR-PR. These lines can contain comments that help explain/document what is being performed in the script.
- Lines may be either tab- or space-separated. For this reason, the separated elements within a line (such as a plate name alias) should not themselves contain tab or space characters (*i.e.*, use “DrinksPlate” rather than “Drinks Plate”).
- Variable definitions (*e.g.*, plate name aliases) must precede their use in the script.

A PR-PR configuration file consists of:

1. A link to robotic table file. In most instances, table files are created using software distributed with the robot. There may also be a set of ready-made table files to choose from.
2. The script itself, consisting of two logical sections: definitions and actions.

Definitions Section

(Experiment) name

The name of the experiment may optionally be specified:

```
NAME      BreakfastDrinks
```

Table (file)

The name of the robotic table file to use must be specified.

```
TABLE      BreakfastDrinks.ewt
```

For the stand-alone version of PR-PR, it is necessary to include the (.ewt) table file in the “Tables” folder inside the PR-PR folder:

Name	
▼	PrPr
▼	Tables
	BreakfastDrinks.ewt



It is very important to verify that the correct table file is specified for the experiment.

Documentation Section

A documentation section, enclosed by a pair of tripled-quotation mark characters ("\""), may be included in the script. This section is operationally ignored by PR-PR (much like lines beginning with the pound character '#'), but can help explain/document what is being performed in the script.

Here is an example documentation section:

```
"""
Recipe for breakfast drinks.
"""
```

Plate (name aliases)

Aliases may be specified for the plate names (that are themselves specified in the robotic table file):

#	alias	name
PLATE	DrinksPlate	PL4



Note: plate name aliases must not contain any space characters.

Locations (sources and destinations)

Locations specify plates and wells. Wells are specified as follows:

1. For a single well: by letter-number notation (*e.g.*, "A1") or by number-only notation (*e.g.*, "1").
2. For multiple consecutive wells: the first well, then the character '+', and then the total number of wells. For example, "A1+4" indicates the four consecutive wells starting from well A1 (*i.e.*, wells A1, B1, C1 and D1).
3. For multiple non-consecutive wells: the wells separated by commas ',' (*e.g.*, "A1, C1, E1, G1").
4. Consecutive and non-consecutive wells may be interspersed (*e.g.*, "A1+4, F1").

A plate and its wells are separated by a colon ':' (*e.g.*, "PL8:A1+4, F1"). Multiple plates are separated by forward slashes '/' (*e.g.*, "PL8:A1+4, F1/PL5:A1, C1, E1"). Plate name aliases may be used instead of the plate name (*e.g.* "WaterPlate:A1+4, F1" instead of "PL8:A1+4, F1").



Note: locations must not contain any space characters.

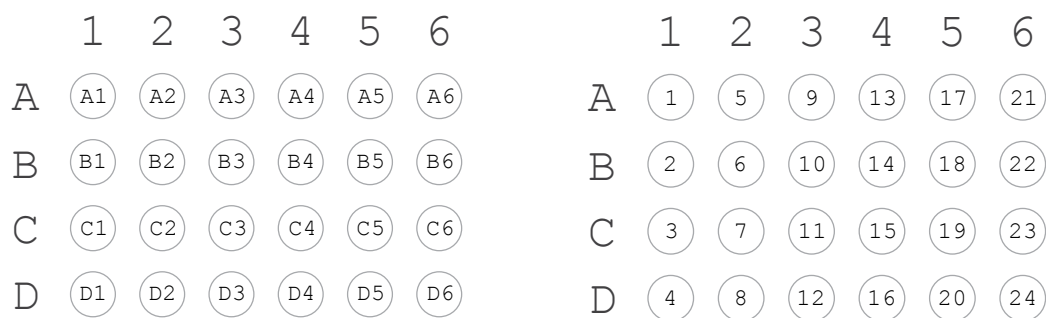


Figure 1. Plate well locations in letter-number notation (left), and in number-only notation (right).

Wells are ordered sequentially first from top to bottom, and then left to right. 'A1+4' (or '1+4'), indicates wells 'A1, B1, C1, D1' (or '1, 2, 3, 4').

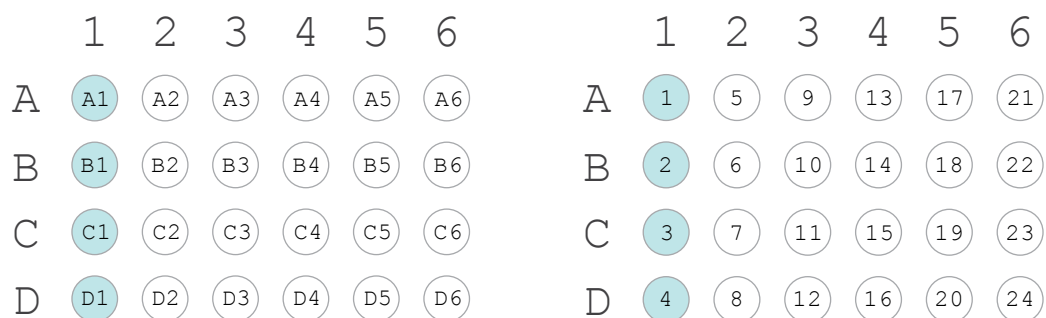


Figure 2. Plate well locations indicated by 'A1+4' (left) or '1+4' (right).

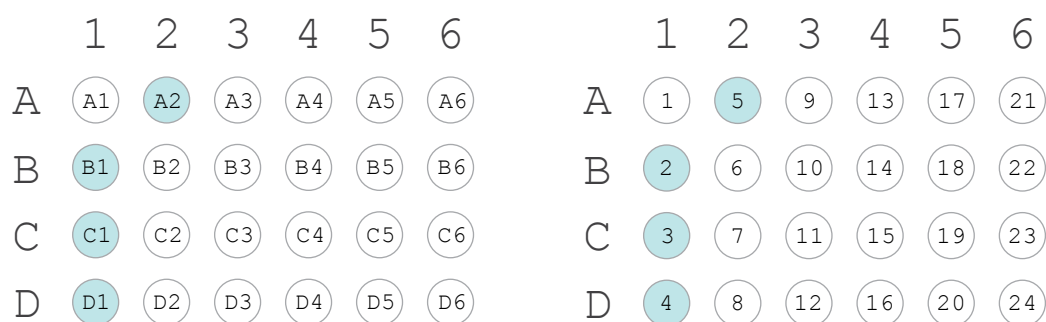


Figure 3. Plate well locations indicated by 'B1+4' (letter-number notation, left) or '2+4' (number-only notation, right). Note that for this particular plate geometry (4 rows by 6 columns), well location A2 (or 5) follows well location D1 (or 4).

Methods

Methods specify the type (or class) of a liquid, as well as the pipetting method for transferring the liquid from one location to another.

The method should be one of the following:

1. `LC_W_Bot_Bot` Water, aspirated from the bottom of the well, dispensed to the bottom.
2. `LC_W_Bot_Lev` Water, aspirated from the bottom, dispensed to the liquid level.
3. `LC_W_Bot_Air` Water, aspirated from the bottom, dispensed in air above the liquid.
4. `LC_W_Lev_Bot` Water, aspirated from the liquid meniscus level, dispensed to the bottom.
5. `LC_W_Lev_Lev` Water, aspirated from the liquid level, dispensed to the liquid level.
6. `LC_W_Lev_Air` Water, aspirated from the liquid level, dispensed in air above the liquid.
7. `DEFAULT` `LC_W_Bot_Bot`; see immediately below.

Outside of `COMPONENT` definition statements (see immediately below), `DEFAULT` refers to the method specified for the component in its definition statement. If there is no method pre-specified for the component, the method defaults to `LC_W_Bot_Bot`.

Custom methods

In the web version of PR-PR, it is possible to specify the additional (custom) methods to use. They are created and used on a per-experiment basis.

Please note, that PR-PR **will not** create additional Liquid Classes in your robot settings; you should do it yourself or ask a specialist. This feature is used to tell PR-PR which liquid classes you have (if their names are different from the default ones), so it could validate against them and correct errors in the user script.

To specify additional methods, click on the 'Setup custom methods' link on the PR-PR webpage.

Welcome to PR-PR!

[Paper](#) :: [Copyright](#) :: [Disclaimer](#) :: [Get code](#) :: [Version 0.33](#)

A new simple way to make your experiments faster.
Simply write your code in the window below, add a table file and click 'Prepare robot file'.
The resulting file will be available for you to download on the right.

 [PR-PR Howto guide \(PDF\)](#)

1. Select output device

Tecan Freedom Evo

[Setup custom methods](#)

2. Select/upload table file

BreakfastDrinks.ewt

[Preview table layout](#)

3. Write script [load example](#)

NAME	BreakfastDrinks
TABLE	BreakfastDrinks.ewt

Any methods you enter in the appeared field will be added to the current experiment along with the ones existing by default. Just type in the method name and click ‘Add new’.

[Setup custom methods](#)

Custom methods

Add new

My_Custom_Method

×

Another_Method

×

The new method will be added to the list. On the screenshot below you can see the list of custom methods that have been created for this experiment.

[Setup custom methods](#)

Custom methods

Add new

My_New_Method

×

My_Custom_Method

×

Another_Method

×

To make one of the custom methods default (the one that will be used for components where no method is defined), click on it once. The star icon and blue color will indicate that the method is set to be default.

[Setup custom methods](#)

Custom methods

Add new

My_Custom_Method

×

Another_Method

★

×

Another_Method

×

To delete a method, click on the cross icon on the right side of the method name.

[Setup custom methods](#)

Custom methods

Add new

My_New_Method

×

My_Custom_Method

×



Component(s)

Components may be specified with names, locations, and pipetting methods:

#	name	location	method
COMPONENT	Water	PL8:A1+4,F1	LC_W_Lev_Air

PL8 Water

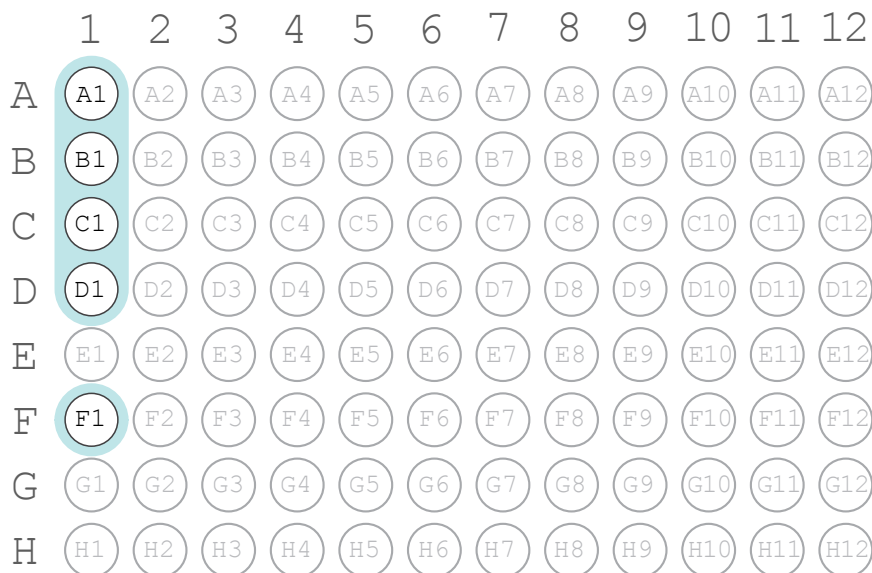


Figure 4. Component location PL8 : A1+4 , F1 on the sample plate (8 rows by 12 columns).



Note: component names must not contain any space characters.

Volume (aliases)

Aliases may be specified for volumes (in μL):

#	alias	volume (μL)
VOLUME	DrinkVol	50
VOLUME	WaterVol	25



Note: volume aliases must not contain any space characters.

Recipe(s)

Recipes may be defined to alias specific mixtures of components, such as a PCR master mix.

The definition of a recipe begins with a `RECIPE` line that specifies the name of the recipe as a whole. The lines that immediately follow specify the names and contents of the sub-recipe(s) associated with the recipe (one sub-recipe per line) . In the example below, the recipe `Drinks` has three associated sub-recipes: `chai`, `coffee` and `lemonade`. Each sub-recipe name is followed by a colon `:`, which in turn is followed by the contents (specified in pairs of component name or locations, and volume (in μL)) of the sub-recipe. Plate name aliases, components, and volume aliases (after they have been defined) may be used when defining a recipe.

```
#          name
RECIPE    Drinks
#          component1  volume1  component2  volume2  component3  volume3
chai:     TeaExtract  30      Syrup       30      Water       WaterVol
coffee:  BeanExtract 30      Milk        30      Water       WaterVol
lemonade: LemonJuice  15      PL7:18      45      Water       WaterVol
```

Actions Section

Options

Options specify what should happen after liquid has been dispensed into a well.

Each option should be one of the following:

1. `MIX:volume×repetitions` Aspirate/dispense *volume* µL *repetitions* times (e.g. `MIX:25×20`)



Note: options must not contain any space characters.

Make (a recipe)

The `MAKE` action prepares a defined recipe, or sub-recipe(s), at the specified location(s).

Sub-recipes are prepared in separate locations, and thus a location must be specified for each sub-recipe. For consecutive well locations, the sub-recipes are prepared sequentially into consecutive wells.

#	recipe:sub-recipe	location	method	options
MAKE	Drinks	DrinksPlate:A6+3	DEFAULT	MIX:25×20
MAKE	Drinks:coffee,lemonade	DrinksPlate:A1+2	DEFAULT	MIX:30×10



Note: `recipe:sub-recipe(s)` must not contain any space characters.

Spread (a component)

The `SPREAD` action distributes a single defined component (or the same liquid present in one or more source locations) to one or more destinations.

#	component	destination	volume (uL)	method	options
SPREAD	Water	PL6:A4+10,A6	DrinkVol	DEFAULT	MIX:25×20
SPREAD	PL4:A1+4	PL6:A4+10,A6	DrinkVol	DEFAULT	MIX:25×20

Transfer (liquids)

The `TRANSFER` action distributes liquids one-to-one from source to destination locations. A destination location must be specified for each source location.

#	source	destination	volume (uL)	method	options
TRANSFER	PL1:A1+3	PL6:A7+3	150	LC_W_Bot_Bot	MIX:15×8

Protocols

Protocols allow reusing of both definitions and actions, when a lot of similar slightly different actions are needed. As recipes, they are first defined and then called. Protocol is located between the commands `PROTOCOL` and `ENDPROTOCOL`. You can use as many definitions and actions as you like.

```
#protocol      name      variables...
PROTOCOL      MyProtocol  MyLocation    MyMethod

#             name
RECIPE        Drinks
#             component1  volume1  component2  volume2  component3  volume3
chai:         TeaExtract  30       Syrup       30       Water       WaterVol
coffee:      BeanExtract 30       Milk        30       Water       WaterVol
lemonade:     LemonJuice  15       PL7:18      45       Water       WaterVol

#             recipe:sub-recipe      location      method      options
MAKE        Drinks                  MyLocation    MyMethod     MIX:25x20
ENDPROTOCOL
```

Protocol is then called by a command `USE`. The amount of values provided with the command should be the same as the amount of variables in the protocol definition.

```
#use      protocol_name      values...
USE       MyProtocol         PL4:A1+3      LC_W_Lev_Bot
```

Variables and their values are matched one-by-one, left-to-right:

```
#protocol      name      variable1      variable2      ...
PROTOCOL      MyProtocol  MyLocation     MyMethod
#use          name      value1         value2         ...
USE           MyProtocol  PL4:A1+3       LC_W_Lev_Bot
```

One protocol can be used more than once with different variable values.

```
#use          name      value1      value2
USE           MyProtocol  PL4:A1+3    LC_W_Lev_Bot
#use          name      value1      value2
USE           MyProtocol  PL6:B3+3    DEFAULT
```

Putting it all together: an example PR-PR script

```
NAME      BreakfastDrinks
TABLE     BreakfastDrinks.ewt
```

```
""
Recipe for breakfast drinks.
""
```

```
#      alias      name
PLATE  DrinksPlate PL4
```

```
#      name      location      method
COMPONENT Water      PL8:A1+4,F1  LC_W_Lev_Air
COMPONENT TeaExtract  PL7:17      LC_W_Lev_Bot
COMPONENT Syrup       PL7:18      LC_W_Lev_Bot
COMPONENT Milk        PL7:19      LC_W_Lev_Bot
COMPONENT BeanExtract  PL7:20      LC_W_Lev_Bot
COMPONENT LemonJuice   PL7:21      LC_W_Lev_Bot
```

```
#      alias      volume(uL)
VOLUME DrinkVol    50
VOLUME WaterVol     25
```

```
#      name
RECIPE Drinks
#      component1  volume1  component2  volume2  component3  volume3
chai:   TeaExtract  30      Syrup       30      Water       WaterVol
coffee: BeanExtract  30      Milk        30      Water       WaterVol
lemonade: LemonJuice  15      PL7:18      45      Water       WaterVol
```

```
#protocol      name      variable1      variable2      variable3
PROTOCOL       MyProtocol MyComponent     MyLocation     MyMethod
#protocol_details
SPREAD         MyComponent     MyLocation     40             MyMethod
ENDPROTOCOL
```

```
#      recipe:sub-recipe      location      method      options
MAKE   Drinks                DrinksPlate:A6+3  DEFAULT     MIX:25x20
```

```
MAKE   Drinks:coffee,lemonade  DrinksPlate:A1+2  DEFAULT     MIX:30x10
```

```
#      component  destination  volume(uL)  method      options
SPREAD  Water      PL6:A4+10,A6  DrinkVol    DEFAULT     MIX:25x20
```

```
#      source      destination  volume(uL)  method      options
TRANSFER PL1:A1+3      PL6:A7+3    150         LC_W_Bot_Bot  MIX:15x8
```

```
#use      name      value1      value2      value3
USE        MyProtocol  Milk        PL4:A1+3    LC_W_Lev_Bot
```