New step for OC_manager

Implementing a new step for OC_manger can be done without touching the main code by adding files in different folders.

This document describe the procedure, knowledge of GCODE and R is necessary.

http://reprap.org/wiki/G-code

If the step can be done by a succession of GCODE and options can be supplied in a simple table, a step can be created.

As example, a step of plate heating will be implemented.

tables folder

In the table folder, a CSV file must be created with the following constraints:

- The separator is semi-colon.
- The name must finish with **.csv**.
- Column names must be **Option** and **Value**.
- Row names must be present, the names have no influence though and are here to inform the user.
- The **Option** column contain the options names that the step function will later catch (more later).
- The **Value** column contain the default values for the options.

For plate heating, two options are needed: temperature and time. The file will look like that:

```
Option;Value
Temperature (°C);Temperature;100
Time (min);Time;10
```

Other files for inspiration can be found in GitHub.

https://github.com/DimitriF/OC_manager/tree/master/tables

eat_tables folder

In the eat_tables folder, a R script must be created:

- This R script contain a function that will take a step list and update it, *i.e.* modify the `gcode`, `info` and `plot` elements.
- The R script must have the same name as the CSV file
- The R script must return the step object

For plate heating, here is what it looks like:

```
# eat_table_heating
function(step){
  # eat_table_heating function V011,24 february 2017
  # home, start heating, and wait the needed time, then home again

  # extract the needed information
  table = step$table
  Temperature=table[table[,1] == "Temperature",2];Time=table[table[,1] == "Time",2]

  # make the gcode
  gcode = c("G28 X0; home X axis",
            "G28 Y0; home Y axis",
            paste0("M190 S",Temperature," ; set temperature, wait to reach, use M140 to set and go to t
            paste0("G4 s",Time*60," ; time wait in secondes"),
            "M190 S0 ; set temperature off",
            "G28 X0; home X axis",
            "G28 Y0; home Y axis"
  )

  # make the new plot
  plot_step=function() {
    plot(c(1),c(1),type="n",main="No plot for heating step")
  }

  # replace the elements in the list
  step$gcode = gcode
  step$plot = plot_step
  step$info = paste0("Heated as at ",Temperature, "°C for ",Time*60, "sec")
  return(step)
}
```

Other files for inspiration can be found in GitHub.

https://github.com/DimitriF/OC_manager/tree/master/eat_tables

Server side

Everything happens in the server_Method.R file.

When the web page is loaded, the server will look inside the `eat_table` and `tables` folder to store the list of available steps and let the user select among them for a method creation.

Method_step_add

When the user click on the + button (`input$Method_step_add`), the selected step is added to the Method list. This step is also in the form of a list with different element, *i.e* `table`,`eat_table` (the function we created),`plot`,`gcode` etc...

Method_step_update

When the user update the step (`input$Method_step_update`), the step is fed to the `eat_table` function.

```
Method$l[[step]] = Method$l[[step]]$eat_table(Method$l[[step]])
```

Method_step_exec

When the user execute the step (`input$Method_step_exec`), the gcode is written in the file `gcode/Method.gcode` and launch (`main$send_gcode(Method_file)`).

Additionnaly, it is possible to delete a step and also save and load the method as a whole for later use.

UI side

On the UI side, still in the server_Method.R file, the user select a step with the radioButton `input$Method_steps`, the server will render tables, plot, gcode and info corresponding to this step. When one of the button will be clicked, it will just concern this step.

Special case of appli_table

In some case, a complementary table is needed for a more granular control of the step, *e.g.* sample application, derivatization. The `appli_table` is not present by default in the step and must be added to the list when the update take place. The UI will then be able to detect the presence of the `appli_table` in the step list and display it for further editing and use during the update.

Special case of the Documentaion step

The `Documentation` method is special because it need to use the shell to take pictures with the rpi camera. It is still in the Method module but need a complementary file (`Documentation_exec.R`) to be executed, in this case, an other object `exec` is added to the step list. It also need an appli_table to choose light , exposure time and ISO speed (even though this is not an application anymore, the name `appli_table` had been kept).