Guide to Setting up the Windchime Simulation

and Analysis Operating Platform

There are two main stages to setting up the operating platform for the Simulation and Analysis frameworks of Windchime. Those are 1) setting up the import packages and 2) setting up the local config and json files. I will go over each here.

---------------------------------------------------------------------------------------------------------------------

There are two ways to setup the import package list, pick whichever one depending on your preference.

*Option 1 (Recommended):*

This option is recommended because, other than all the imports in the example notes being set to this setup, the import packages are also connected to github so that any changes to the master will be reflected into your individual repository, removing any need to download from github again (instead requiring just a pull request).

1. Figure out where your IDE or operating platform looks for its import packages
   * This is a helpful source to figure that out: <https://stackoverflow.com/questions/122327/how-do-i-find-the-location-of-my-python-site-packages-directory>
   * In general, in the command line, run: python -m site
   * One of these sites is the folder that has the file(s) importing the packages (the local directory). This is fine, but not recommended, to use because it will be local, not global (any files in another location will not import from it implicitly).
2. Pick one of these locations and clone the two github repositories, ‘planck-mass-MC’ and ‘planck-mass-Analysis’, to this location using github desktop or a similar application.
3. Setup Symlinks for your operating system. You can do this by either:
   * Installing it from a GUI command line (works for linux and for windows 10 if upgraded to pro version, not the home version… or for the home version with a specific external download – link if interested: <https://www.itechtics.com/easily-enable-group-policy-editor-gpedit-msc-in-windows-10-home-edition/>, too complicated though)

Or,

* + Installing it from Shell using the procedure from: <https://schinagl.priv.at/nt/hardlinkshellext/linkshellextension.html>
    1. Note: for this, you will need to have installed chocolatey

1. Go inside the ‘planck-mass-MC’ repository in your chosen location, right click on ‘planckmc’, and choose ‘Pick Link Source’ (this is enabled by Symlinks).
2. Go back to your chosen location (one file up), right click anywhere, hover over ‘Drop as…’, and pick ‘Hardlink Clone’.
3. Go inside the ‘planck-mass-Analysis’ repository in your chosen location, right click on ‘planckanalysis’, and choose ‘Pick Link Source’.
4. Repeat step 5.
5. The package list is now set.

*Option 2:*

This option is similar to the first one but does not have a link to the github repository.

1. Figure out where your IDE or operating platform looks for its import packages
   * This is a helpful source to figure that out: <https://stackoverflow.com/questions/122327/how-do-i-find-the-location-of-my-python-site-packages-directory>
   * In general, in the command line, run: python -m site
   * Again, this can be the local directory.
2. Pick one of these locations.
3. Download the two repositories from the online github source and either:
   * Extract the folders from their individual zipped files, place both in the chosen location, and return to Option 1, step 3 and continue… If this is what you do, consider yourself as following Option 1 **not** Option 2 (for later), with the only difference being that there is now no connection to github.

Or,

* + Place ‘planckmc’ from ‘planck-mass-MC’ and ‘planckanalysis’ from ‘planck-mass-Analysis’ into this chosen location. Place ‘generate\_linear\_response\_json’ from ‘planck-mass-MC’ into the ***local directory*** (i.e. where the file(s) will run, irrespective to where the chosen location is). Following this step is Option 2.

1. The package list is now set.

---------------------------------------------------------------------------------------------------------------------

To set up the local config and json files

1. Copy ‘config’ (.ini file, *important*) from ‘planck-mass-MC’ into your local directory (where you will run any simulation or analysis framework code).
2. Copy ‘Position\_update’, ‘Characteristics\_update’, and ‘Response\_update’ from ‘Documentation’ in ‘planck-mass-MC’ into your local directory.
   * Note: If you followed Option 2, then copy ‘Response\_update\_special’ from ‘Documentation’ instead of ‘Response\_update’
3. Run ‘Position\_update’ ***first***. Make sure the settings are set as desired. Make sure the version matches with the version in Config.
   * **Do Not** change the version number **except** to denote a change in the other setting such as, for example, a different detector structure. If there is a discrepancy and you have not made such a change, then attempt to locate the correct config file. If the version number was to change, then change it in the settings in ‘Position\_update’ and in config.
   * Follow the instructions for the settings to set detector sensor positions as desired (forming different structures and shapes, number of sensors, etc.)
   * The result is a json file with the positions of the sensors
4. Then run ‘Characteristics\_update’ and ‘Response\_update’ (or ‘Response\_update\_special’). The results are the characteristics json file, which includes the sensor orientations, sensitivities, and noise, and the response json file, which includes the linear response (FIR filter) and the signal transfer response (adc lookup dictionary for the accelerations).
   * Note on ‘Response\_update’: like ‘Position\_update’, there is a setting to change here, and that is the Resp variable being sent into the called function. The set default is an FIR filter of length 841 semi-modeled after the actual sensors in consideration (details are in the function of the same name within generate\_linear\_response\_json), but the filter can be something else. Currently there are two more test options, but more can be added following the same format by adding FIR filter functions to ‘generate\_linear\_response\_json’ and adding their name options (in the same format) to the ‘output\_response\_json’ function in ‘generate\_linear\_response\_json’.
   * Note on response.json in general: ‘Response\_update’ (and ‘Response\_update\_special’) can be foregone completely by running ‘generate\_linear\_response\_json’ from the command line, which is a feature provided by that module. The steps here (for response) are just to illustrate another method. The Position\_ and Characteristics\_ updates would still be needed, though.
5. The Simulation and Analysis frameworks are now ready to be run. For example simulation code (animation, graphic, and acceleration analysis), access ‘planck-mass-MC/Documentation/Examples’. For example Analysis code (single variable analysis, multi-variable analysis, etc.), access ‘planck-mass-Analysis/Analysis-Notebooks’.