NixLyn_Lab (pip package):

In order to achieve a large PyPi package, it would be best to first create smaller packages which can then be imported into other packages.. (nested packaging), an example (and first place to start) would be:

File_man()

From the first file_man.py I created, I've copied, pasted and improved the 'lib tool' to the point where it became part of every project.. however, it has become quite a hassle to "lug around". Currently File_man functions require exact arguments and are mainly focused on str and list types..

Tools (functions) include:

- read_file(file_name, delim)

-> reads the file content as a str
-> returns a list seperated by the given delim

- write_file(file_name, data, delim, rwm)

-> checks for some bad chars in the filename
-> checks if type(data) is;
--> list (adds each item+delim to the data_str)
--> str (data_str = data)
-> opens file according to the rwm["w", "w+", "a", "a+", "x"]

- check_file(path_+file_name)

-> return True is file exists
-> return False is not

- check_file(path_)

-> return True is path_ exists
-> return False is not

- file_list(path_)

-> if_path: return list of all files in path_

- make_dir(path_)

-> creates directory(path_)

Future Implementation:

- write_file(*args):

->Args[index] - value type eg

--> `fm.write_file(fname="file.txt", data_=data, delim="\n", stat=True)`

  * fname is required
  * if ! data_ -> empty file is made
  * if ! delim -> no delim (or according to data type: if eg map, set, tuple, etc)
  * check for data_types: str, list, map, set, tuple, bytestr
  * stat (default=False), if True -> uses tqdm

- read_file(*args):

->Args[index] - value type eg

--> `fm.read_file(fname="file.txt", rtype="list", delim="\n", stat=True)`

  * fname is required
  * if ! rtype, && ! delim -> return str
  * if ! rtype, but delim -> return list
  * if rtype, delim=void -> return rtype(data)
  * stat (default=False), if True -> uses tqdm

- get_tree(path_, rtype, stat)

-> return entire directory tree of path_
-> rtype options => [nested lists, map] (default=nested list)
-> if stat=True => tqdm will be implemented

eg.
Tree:
```
path_/
|__cpath_0/
|    |__cpath_0_a/
|    |    |__file_0_a_x.txt
|    |__file_0_a.txt
|    |__file_0_b.txt
|__cpath_1/
|    |__file_1_a.txt
```

List:
```
["path_/",
    ["cpath_0/",
        ["file_0_a_x.txt",
        ],
        "file_0_a.txt",
        "file_0_b.txt",
    ],
    "cpath_1/",
        ["file_1_a.txt",
        ],
    ],
]
```

> The "get_tree()" function will include the files, as well as there data..
> The "show_tree()" will simply return the paths+names

- plant_tree(tree_, new_path)

-> I'll need to figure that part out...