Project Eager Mercury

Technical Documentation

GitHub page - https://github.com/alexander-bzikadze/Project-Eager-Mercury.
PivotalTracker page - https://www.pivotaltracker.com/n/projects/1646871.

Our Dictionary:

- Pem - from now on, Project Eager Mercury.
- Current project - project, the user is working with at the current moment.
- Infofile - file, located in (sublime text package folder)/Pem/Info.txt. It contains information of all the projects names and paths. In the first line there is number of current project, current project name, path to the current project. If no project is current, the first line should be «-1\n». In other lines project names and paths to the projects are contained.
- Selected project, file or anything else - given by user (Not to be the same with current).
- Project file - file, that lies in the project directory (directory, that is pointed by project's path). Its extension is .pem. In first line project's name is contained. Then specification list, then source list.
- Specification - under specification list of connected libraries, packages and so on is meant.
- Source - list of .cs files.

List of classes and brief information about them (A Static Structure Diagram is attached):

I. **Kernel (Staff folder)**:
   - InfoReader in Staff/readerWriter.py. Reads information from infofile and then provides following information: current project name, path to current project, list of project names, list of paths to projects (with the same numbering as in previous item), number of project in mentioned list by provided project name.
   - InfoWriter in Staff/readerWriter.py. Writes some kind of information to infofile. Kind of information, that can be written: addition of a new project, deletion of a project by number, switching current project to another.
   - ProjectReader in Staff/readerWriter.py. Reads information from current project's projectfile and then provides following information: projects name, path to the project, get list of specification, get source list.
   - ProjectWriter in Staff/reader.Writer.py. Writes some kind of information to current project's projectfile. Kind of information that can be written: addition of a new file to the project, deletion of a file from the project.

II. **Command classes** (all these classes contain one public method - run method):
   - AddFileCommand in addFile.py. Creates a new template .cs file and adds it to the current project. Run takes «name» argument.
   - CompileProjectCommand in compileProj.py. Compiles current project and prints all the information received from the console. Works like that: reads information of current project with the help of ProjectReader, creates a Makefile with a goal to compile the project, executes it, deletes it. Run takes «target» argument, that defines kind of binary file (0 - .exe, 1 - .dll, 2 - .winexe, 3 - .netmodule).
   - RunProjectCommand in runProject.py. Finds .exe of the following kind: «project name» + .exe. If found, executes it and prints all received from the console information. As CompileProjectCommand, works with Makefile.
   - CreateProjectCommand in createProject.py. Adds information to infofile of a new project, if successful, creates a template projectfile with provided name in provided directory. Run takes «name» and «path» arguments. If path not given, it is considered users directory.
   - DeleteFileCommand in deleteFile.py. Removes information from the current project's projectfile about selected source file. If successful, deletes the file. Run takes «name» argument. Run takes «name» argument.

- DeleteProjectCommand in deleteProj.py. Removes information from infofile about selected project. If successful, deletes all source files, connected to the project, then deletes projectfile.
- GetAllProjCommand in getAllProj.py. Prints list os all project, stored in infofile.
- GetFilesCommand in getFiles.py. Prints list of all source files, connected to the current project.
- InfoCommand in Info.py. Prints name of and path to current project.
- SwitchProjectCommand in switchProj.py. Switches current project in infofile to selected. Run takes «name» argument.
- OpenFileCommand in openFile.py. Opens in Sublime Text selected file. Run takes «name» argument.
- OpenProjectCommand in openProject.py. Opens in Sublime Text all source files, connected to selected project. Run takes «name» argument.
- Pem in pem.py. This command is special. Its purpose is to run other commands. Run takes «args» argument. If given, runs selected in args command. If not, opens input_panel and takes command name from the user. Then, if necessary, opens input_panel to take arguments from

III. **JSon files**.

- Context.sublime-menu. Places per command and Folder «Pem Commands» with other commands to right button context menu.