**CommandExecution Class**

**Description**

This class is used to take in input parameters by users from the UI and analyze the input so as to perform respective functions called by users. So far, there are five primary functions that can be called by users and they are “add”, “delete”, “display”, “update” and “exit”.

Whenever users call one of the five main functions, this class will call Interpreter class which is the private attribute “inter” to convert the user input to standard codes which will be verified by StandardCommand and stored correct data in TaskInfo taskInfo for subsequent uses. Finally, the right data that users want to store in the local storage file will be stored in tasks (i.e. a storage class).

There are four private attributes: string \_content, Interpreter inter, TaskInfo taskInfo and Storage tasks.

The \_content is used to store the remaining input after the user performs a command. For example, if the user input “delete 1”, the \_content will be “1”. The inter is the object of the Interpreter class. The taskInfo is a local Struct which is used to store single pieces of data that a user wants to add to local file. The tasks is the object of the Storage class.

**General function**

string CommandExecution::readCommand(string userInput) {

size\_t end=userInput.find\_first\_of(" ");

string command=userInput.substr(0, end);

\_content=userInput.substr(end+1, userInput.size()-end);

command=inter.interpretCommand(command);

string message="";

executeCommand(determineCommandType(command),message);

return message;

}

In this readCommand function, the user input will be passed from the UI as a string parameter, and this function will cut the user input to two parts. The first part is a single word which represents the command (i.e. “add”, “delete”, “display”, “update” and “exit”) called by the user. It is stored in the string command, and this function will call the Interpreter class to convert the word to standard code for subsequent analysis (i.e. determineCommandType). The second part is the remaining part of the user input which will be stored in the string \_content. This \_content is a private attribute which will be used by other public functions in later analysis. Moreover, the string message will be continuously passed by reference to subsequent functions and returned to this readCommand function. Lastly, the message will be returned as a string to the UI to display corresponding text result to the user.

**Primary function**

After a command type is determined in the readCommand function and passed to the executeCommand function, one of the five primary functions will be called. One of the five primary functions is the performAdd function. It will take a string& message which represents the result after performing this function and passed back to the UI in the end.

void CommandExecution::performAdd(string& message) {

inter.convert(\_content);

storeInTaskInfo();

addEventToList();

message=addResult();

}

In this function, firstly, the Interpreter class (i.e. inter) will be called to convert the user input to standard format for subsequent analysis and stored in the string \_content. Secondly, the \_content will be cut to smaller parts where each part represents one piece of information (i.e. “month”, “day”, “startTime” and etc.) and stored in the TaskInfo tasInfo. This is done through calling storeInTaskInfo. Thirdly, the Storage class will be called (i.e. addEventToList()) to store every piece of information to local file in a right format. Lastly, the message will be assigned with corresponding text result (i.e. string) and returned back to the UI in the end.