Table of Contents

Overview of Program

This program is modeled after the command design pattern. The principle elements of the system is the Command, Receiver, Concrete Command classes, and Invoker classes. The actual command functions are all contained within Receiver.

Program Structure

The program file (OS15.cpp) contains the Command, Receiver, the Concrete Commands (which are the SeeVersion, SeeDate, SeeDirectory, ExitSystem, and SeeHelp classes), and the Invoker classes.

The Command class includes:
1. The execute function which handles the actual execution interface of the system.

The Receiver class includes the actual commands as follows:
1. The version function which displays the current version of the system.
2. The date function which displays the current time and date.
3. The directory function which calls upon the file tree walk (ftw) function to retrieve a list of files in the current directory.
4. The dir function which actually displays the directory file list based on what is received from the directory function.
5. The exitSystem function which handles the procedures for shutting down the system.
6. The help function which displays help information for each command contained in separate files.

The SeeVersion, SeeDate, SeeDirectory, ExitSystem, and SeeHelp classes all contain the following method:
1. An execute function which executes the proper Receiver function.

The Invoker class contains the following method:
1. The invoke function which actually executes the command and stores a history of previous commands.

Outside these classes, there is also:
1. A displayMenu function which displays the main menu for the system.

Description of Each Function

The Command class actually defines the interface for executing each command. Below is a description of the commands found within the Command class.

The execute function
Prototype: virtual void execute();
Parameters: This function takes no parameters.

Return Type: This function has no return type.
Description: This function defines the interface for executing the commands.

The Receiver class actually handles the implementation for each command. It is the class that actually does the work. Below is a description of each function found within the Receiver class.

The version function
Prototype: void version()
Parameters: This function takes no parameters.
Return Type: This function has no return type.
Description: This function displays the current version of the operating system using a simple std::cout statement.

The date function
Prototype: void date()
Parameters: This function takes no parameters.
Return Type: This function has no return type.
Description: This function displays the current date at the time the function was executed using the standard ctime libraries.

The directory function
Prototype: void directory()
Parameters: This function takes no parameters.
Return Type: This function has no return type.
Description: This function calls upon the file tree walk (ftw) function to retrieve information on the files within the current directory. The ftw function passes this information to the dir function below.

The dir function
Prototype: static int dir(const char *path, const struct stat *st, int flags)
Parameters: **const char *path –** This parameter denotes which directory this function should print out. In this program, the ftw function in the directory function passes "." as this parameter, meaning it will print only the current directory.
const struct stat *st – This parameter holds which directory needs to be printed out via the ftw function.
int flags – This parameter denotes how many directories should be held open by ftw at once.
Return Type: This function returns type int. It returns 0 if everything executed correctly.
Description: This function prints out the path that is passed into the function by the ftw function inside the directory function using std::cout.

The exitSystem function
Prototype: void exitSystem()
Parameters: This function takes no parameters.
Return Type: This function has no return type.
Description: This function asks the user if they want to shut the program down. If yes, this function calls exit(0) to terminate the program. It not, the function returns the user back to the program's main menu.

The help function

Prototype: void help()
Parameters: This function takes no parameters.
Return Type: This function has no return type.
Description: This function loads the help system main menu from the file helpmain.txt. The user then inputs which command they need help with and depending on their choice, will load the help text from version.txt, date.txt, directory.txt, exit.txt, or help.txt. It will then return the user back to the main menu.

The SeeVersion, SeeDate, SeeDirectory, ExitSystem, and SeeHelp classes are all concrete command classes. The concrete command classes contain the following methods:

The execute function
Prototype: void execute()
Parameters: This function takes no parameters.
Return Type: This function has no return type.
Description: This function executes the proper Receiver function based on which command is passed to it.
Cross References: This function calls upon the proper Receiver function depending on which concrete command is executed.

The Invoker class initiates the command by passing it to the Receiver. The following method is contained within the Invoker class:

The invoke function
Prototype: void invoke(Command* command)
Parameters: This function takes the following parameters:
Command* command – This parameter receives whichever command is input by the client.
Return Type: This function has no return type.
Description: This function executes the proper Receiver function based on which command is passed to it by the client through the Command* command parameter.
Cross References: This function calls upon the execute function in the Command class to initiate the execution of the proper command.

The displayMenu function
Prototype: void displayMenu()
Parameters: This function takes no parameters.
Return Type: This function has no return type.
Description: This function displays the main user interface for the operating system.
Cross References: Depending upon which choice the user chooses, the commands call upon the invoke function in the Invoker class and passes a concrete command object to it.

Alphabetical Index