**CSE 308  
FileManager Project by**

**Saurav Gupta  
1405122**

1. **CLASSES**
2. **FileManager**

This is the main class. This class contains the necessary Graphical User Interface codes and functions and creates objects of necessary classes.

**Methods**:

* **getGui:** This method creates different panels, buttons and other things that are required for UI. This method returns a JPanel.
* **showRootFile:** This method simply just selects the first and only one element in the tree.
* **LoadPath:** This method takes path of the file (in form of string) as argument and sets the data in table and list according to the path.
* **setFilePath:** This method sets the path of the file in the JTextfield.

1. **TreeView**

This class shows the tree and its files. A node is created and then the files present in the node are added to the parent node and displayed in the treeview.

**Methods:**

* **TreeView:** This constructor method initializes and shows the tree with a node and the files in that node.
* **showChildren:** This method shows all the files in the selected node.

1. **TableView**

This class initializes the table and sets the data in the table.

**Methods:**

* **TableView**: This constructor method initializes the table.
* **setTableData**: This method takes array of files as argument and sets these files in the table.

1. **ListView**

This class initializes the list and sets the data in the list.

**Methods:**

* **ListView:** This constructor method initializes the list.
* **setListData:** This method takes array of files as argument and sets these files in the list.

1. **ChooseView**

This class sets the table view or list view accordingly by creating the TableView or ListView objects according to the flag value.

**Methods:**

* **ChooseView:** This constructor method takes the value of flag in integer.
* **chooseviewtype:** This method takes array of files as input and creates and sets the data in table or list according to the flag.

1. **FileRenderer**

This class is for the style and the way of viewing the list items in the listview.

**Methods:**

* **getListCellRendererComponent:** This method looks after how the items will be viewed in the listview. It looks after the alignment of text and icons and other things.

1. **FileTableModel**

This class sets the names of the columns, the alignment of texts and icons and other necessary details required in the table.

**Methods:**

* **getValueAt:** This method takes row and column as arguments and returns the required type of data accordingly.
* **getColumnCount :** This method returns the number of columns in the table.
* **getColumnName:** This method returns the name of column taking the column as argument.
* **getRowCount :** This method returns the number of files in the table.
* **getFile:**  This method returns the file in the given row given as argument.
* **setFiles:** This method just simply sets the current files to be the files as given by the argument.

1. **FileTreeCellRenderer**

This class is for the style and the way of viewing the items in the treeview or in the tree.

**Methods:**

* **getTreeCellRendererComponent:** This method looks after how the items will be viewed in the treeview. It looks after the alignment of text and icons and other things.

**B) DESIGN PATTERNS**

I have tried my best to use all the four design patterns in the FileManager project.

1. **Factory Pattern**

This design pattern is used in ChooseView class where the instances of TableView or ListView can be created according to the flag(integer) value and hence the data can be set in table view and list view accordingly.

1. **Composite Pattern**

This design pattern has been used in the TreeView class. A node is created and then the files present in the node are added to the parent node and displayed in the treeview.

1. **Singleton Pattern**

Here, the class ChooseView can be considered to have the Singleton design pattern because only one object of it is being created.

Also JTextField class (JAVA API ) also can be considered to have the Singleton design pattern because only one object of it is being created.

1. **Adapter Pattern**

Three classes have used the idea of Adapter Pattern.

FileRenderer class

FileTableModel class

FileTreeCellRenderer class

The main aim of these classes is to arrange the texts, icons, images, data in the way Sir is expecting or in the way as shown in sample. It changes the style of viewing the tree, tables, and list in a different way as per the expectation of the teacher.