

Documentation and Design Pattern Usage on *“File Explorer”*

Muhtasim Ulfat Tanmoy
1405086

April 28, 2017

1 Design Pattern Identification and Selection

The given file Explorer project has following requirements:

- **TreeView** - for showing file hierarchy.
- **TableView** - for showing file in a table.
- **TileView** - for showing file in a tiles.
- **Switching** - convenient way for transition between views.

Now, **TreeView** uses the composite design pattern. As we can create different node and each node can also have other nodes as child it matches all the specification for **Composite design pattern**.

TableView and **TileView** has list of files that has been adapted to the view in our code through fileModel class (works as adaptee), this clearly represents the **Adapter class pattern**.

Switching between views like making the same set of files appear in different form in front of users clearly references **Factory design pattern**.

So, TreeView class in javafx implements the composite pattern, the TableView class does the adapter pattern and different view switching implements the factory design pattern. The main function that is initialized only once during the lifetime of the app implements singleton design pattern.

2 Design Pattern Usage

The design pattern usage on our project is described with relevent classes below:

The Main class runs only once and it sets the fxml in the app.It sets the resolution and therefore implements Singleton Pattern.

The Controller class controls the whole app with different properties.

The Treeview API is used here to show the hierarchy of folders to navigate to.This treeview implements composite design pattern.

The **myTreeItem** class implements all the functionalities of a single tree node or element.It's onClick EventHandler implements what to do when an item is clicked.The folder icon is changed when a tree node having children is expanded or closed.And also a list of all available children including folder and file is shown

Now there are two ways of showing the all the file list in the current working directory.

Tableview : Here all the files are adapted to Tableview through file-Model class and CellValueFactory.

The **FileModel class** extends the File class API and relevent file values are filtered throughh this class to our Tableview class.

Tableview : Here all the files are adapted to Tilepane view through TileItems.Here each tileItems are actually a VBox API class.

TileItem : class extends VBox and defines an structure for each item in our tileView.It autometically sets image for file or directory.It also implements **onClickListener** for easy transition to different directories.

The above two methods implement Adapter class

Switching View : Done through ViewFactoryPattern Class.This class takes the view and sets them accordingly.ToggleGroup[has been used for switching views.

3 Design Pattern Implementation

The design pattern implementation is provided in code.

Back functionality is used to to upper hiererchy.

User can also go to differnt folders by typing the path to that folder.

And a meaningful function name has been used and proper documentation is given as comment beside every function and variable declaration.