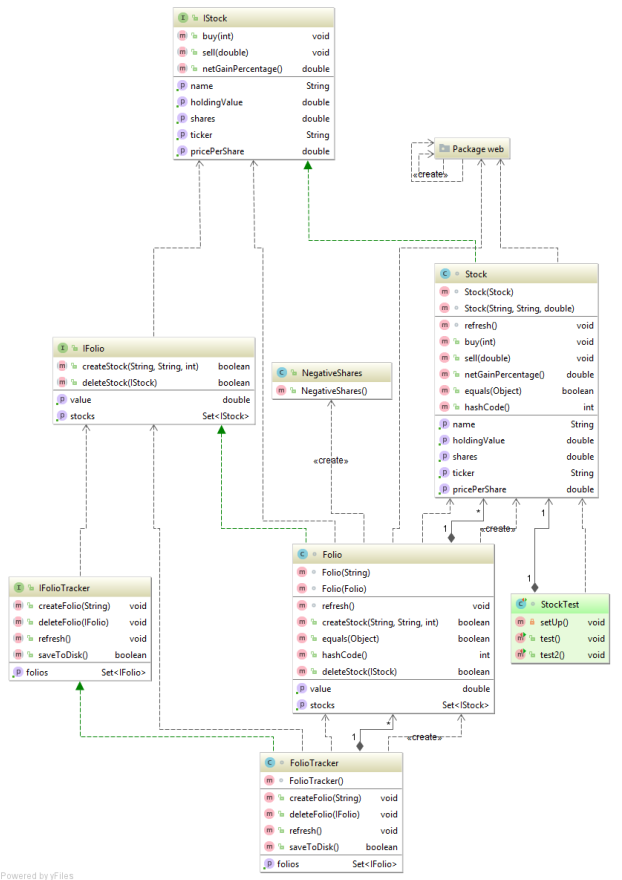
11/20/2017

T06-FolioTracker

Initial Class Diagram and GUI Representation

Steven Barry Cameron Lamberth Lauren Higgins Ioan Luca Chadha Degachi

# 1A) Model Package Class Diagram



# 1B) View Package Class Diagram

# 2.1 Role of Each Interface and Class

Model Package:

IFolioTracker: is there to provide an interface for the FolioTracker class. Its purpose is to provide the method signature of the folio tracker itself. When implemented the code will create and get folios, delete folios, refresh the GUI of the most current up to date stock prices of the NASDAQ stock exchange and also save the details to disk.

FolioTracker: implements the IFolioTracker interface, contains a hash set of different folios to be kept track of and within each individual folio it will keep track of stocks etc. Uses the set collection to get existing folios and create new ones initiated by the user from the GUI.

IFolio: provides an interface which the IFolio class implements. Creates the method signature of add and delete stock, get stocks and get value of stocks.

Folio: Contains a set of stocks used to get the value of the stock at the current time, uses the collection to delete an entry of stock in the set. Overrides both the equals and the hashCode methods, this will be useful for JUnit testing.

IStock: The purpose of this class will be to contain data specific to each stock, this will include details of the ticker symbol, the name of the stock, and the shares bought of each stock and the stock price, as well as the net gain percentage.

Stock: contains a few accessor methods to return the values of the ticker, name, shares etc. also keeps the data encapsulated and hidden. Implements the buy and sell stock methods which decrease or increase the amount respectively.

NegativeShares: This class extends exception, only contains a constructor which is a call to the exception superclass constructor, and will be used to throw a runtime exception when required.

View Package

GUI: Creates the initial user GUI when the application first starts executing, this contains the main jPanel, and has a drop down list for the user to create open save or exit from the folio tracker.

GUIRunnable: implements the runnable interface and creates our implementation of the run method, it has an auto refresh time of 5 seconds and puts the thread to sleep for this specified time.

CreateListener: when the user selects the create option from the main GUI, the user is presented column names of the folio, names such as ticker symbol, stock name, no. of shares, price per share and value of holding, this will give the user the details of each folio. There is also an option to add a new folio which lets them enter the ticker symbol, the share name and the number of shares.

DeleteShareListener: when the user adds new shares to the interface they also have the option to delete, which can be done by right clicking and selecting delete from the menu that appears on user click.

EditShareListener: Creates a jPanel with the title “Edit”, lets the user close the application on exit.

ExitListener: this class will let the user exit the application, when the user selects exit the application shuts down.

OpenListener: when the user selects “open” from the drop down menu from the main GUI, an open dialog is output to the user with a listing of the directories to choose from using a jFileChooser object.

RightClickRow: after the user input the ticker symbol, share name and the number of shares the data is then stored as a row which the user can then right click to either edit or delete, this is implemented within this class.

SaveListener: an open dialog is output to the user with a listing of the directories to choose from using a jFileChooser object and will save details when completely implemented.

View: extends JPanel class and implements the observable interface. This class is used to repaint the GUI and update upon data alteration.

# 2.2 Relationships between Interfaces and Classes:

View Package:

GUI: starts the new GUI thread which then instantiates the run method in the GUIRunnable class.

The GUI also creates the CreateListener, OpenListener, SaveListener and ExitListener classes, these classes contain the code necessary for handling user events which are mainly clicking items from the drop down menu provided and having date in the JPanel as part of the GUI.

As part of the create listener class the user can right click a stock item input by the user, upon this the user can edit or delete the stock items entered. These are done through the EditShareListener and DeleteShareListener classes.

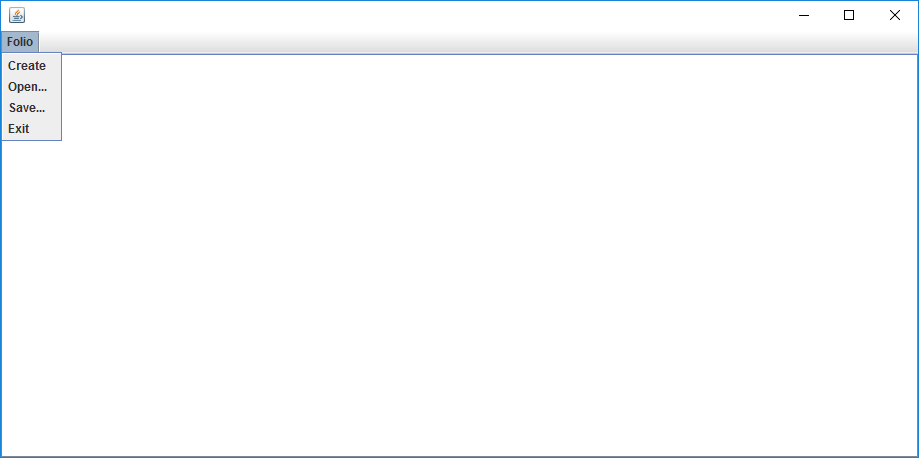
Model Package:

IFolioTracker: is implemented by the FolioTracker class and will provide a means of communication to the GUI through an interface, is also associated with the IFolio and IStock classes as within the IFolioTracker there are many folios which each have to keep track of different stocks within each individual folio. The folio tracker contains a collection (Set) of folios.

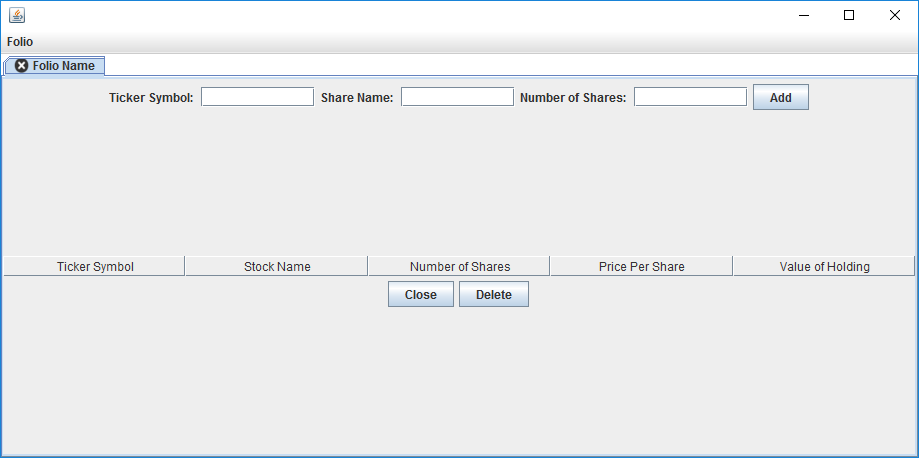
IFolio: implemented by the Folio class and has implementation of letting the user create and delete stock, and refresh the GUI. IFolio is also related to the IStock class which has details of the stock within each folio. The folio tracker contains a collection (Set) of stocks.

IStock: implemented by the Stock class, which contains implementation of buying, selling and the net profit gained or lost from shares of a specific stock.

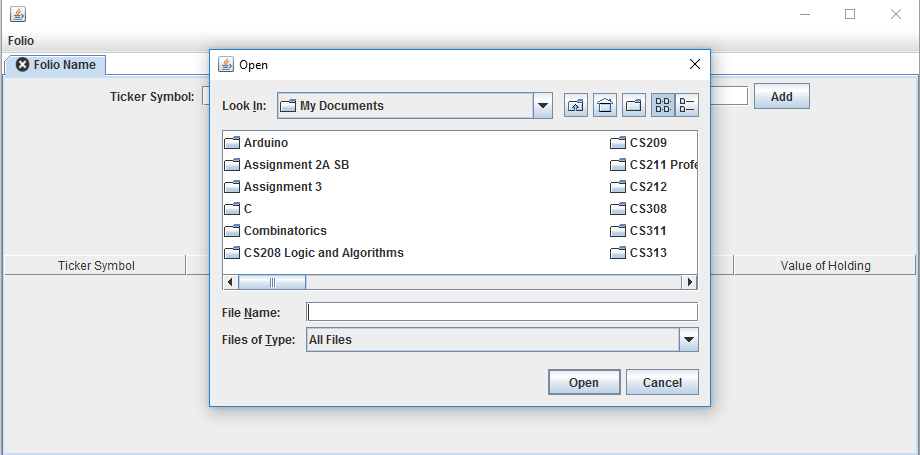
# GUI Representation:



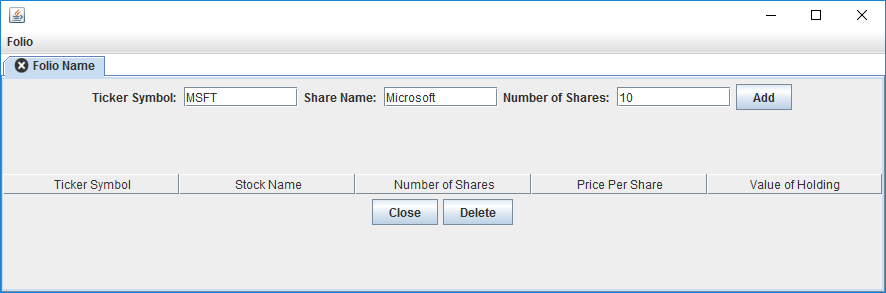
Initial start-up screen with user menu options.



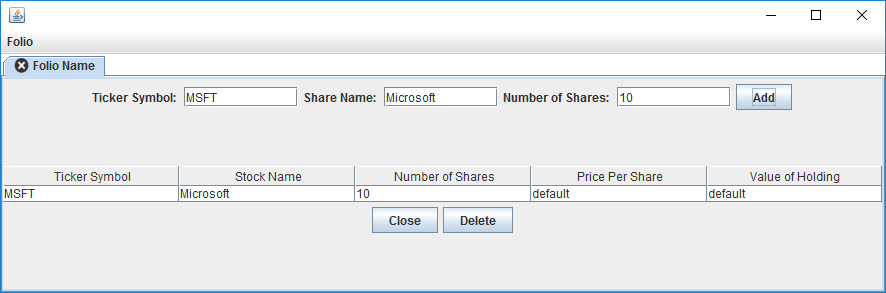
After user clicks the “Create” option the date is added to the JPanel as appears here. The user can then add their own date which has five columns each with details of the shares.



After the user selects the “open” option from the menu, the dialog box appears with a list of directories the user can choose from.



User adding details of shares to be bought.



When the user clicks the “Add” button the details are added to the system and appear as a row as seen here.