CECS 475 – Lab Assignment 2

Sarveshwaran Sampathkumar

Briefly describe how you implement the event. Provide code to declare the events, raise the events and handle the events.

In case of the given assignment, we need to create event whenever the stock reaches its threshold value. The general flow of the program is as follows. The user creates a new stock object. The stock properties are initialized, and the thread is started in the constructor. Activate method updates the value of the stock for every 500 milli seconds using ChangeStockValue method. When the threshold value is reached, a stock event object is created (ACTION delegate), and this event is invoked in the OnThresholdReached method. At this point of time, the publisher publishes this information to all the stock brokers who owns this stock. In the stock broker class, whenever a new stock is added, the stock even handler (listener) is created for that broker for that particular stock. Due to this, whenever the stock reaches the threshold, the invoked event will be handled by the event handler created for that stock for that broker.

Code Snippet:

Creating stock event:

```
if (Math.Abs(currentValue - InitialValue) > notificationThreshold)
{
   if (Program.detaliedLogOutput)
            Console.WriteLine("Threshold reached for Stock : {0}, Initial Value : {1}", this.Name, this.InitialValue);
     StockNotificationEventArgs stockEvent = new StockNotificationEventArgs();
     stockEvent.Name = Name;
     stockEvent.CurrentValue = currentValue;
     stockEvent.NumberChanges = numberChanges;
     OnThresholdReached(stockEvent);
}
```

Invoking the stock event:

```
protected virtual void OnThresholdReached(StockNotificationEventArgs e)
{
   stockEvent?.Invoke(Name,currentValue,numberChanges);
}
public Action<String, int, int> stockEvent;
```

StockNotificationEventArgs Class:

```
public class StockNotificationEventArgs
{
  public string Name {get; set;}
  public int CurrentValue {get; set;}
  public int NumberChanges {get; set;}
}
```

Calling Event Handler:

```
public void AddStock(Stock newStock)
{
  StocksList.Add(newStock);
  stockInitialValueHash.Add(newStock.Name, newStock.InitialValue);
  newStock.stockEvent += Stock_EventHandler;
}
```

Event Handler:

```
public void Stock_EventHandler(string stockName, int currentValue, int numberChanges)
{
  if (Program.firstTimeExecConsole)
  {
          Console.WriteLine(String.Format("{0,-10}\t{1,-10}\t{2,-10}\t{3,-10}", "Broker", "Stock", "Value", "Changes"));
    Program.firstTimeExecConsole = false
  }
          Console.WriteLine(String.Format("{0,-10}\t{1,-10}\t{2,-10}\t{3,-10}", BrokerName, stockName,                    currentValue,
          numberChanges));
    DateTime dt = DateTime.Now;
    String stockDetailedInfo = String.Format("{0,-10}\t\t\t{1,-10}\t\t{2,-10}\t{3,-10}",
    dt.ToString(new CultureInfo("en-US")), stockName, stockInitialValueHash[stockName], currentValue);
    OutputToFile(stockDetailedInfo);
}
```

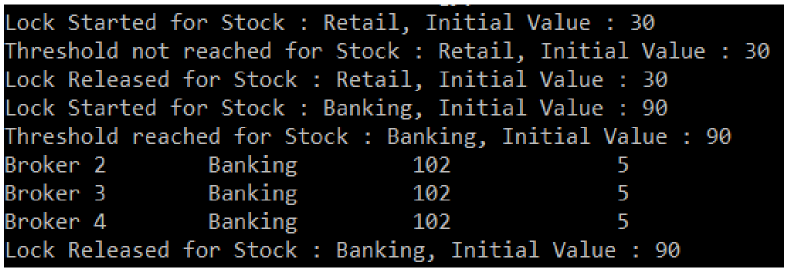
Briefly describe how you handle synchronization. Code to synchronize the outputs to the console window and the text file.

Synchronization plays a very important role in this assignment. This is because, if the process of updating the stock price for a particular task doesn't happen synchronously then there are chances that the updated price of the stock would be incorrect. This is because, the value on which the stock update is happening might be obsolete due to absence of synchronizers. In our assignment, we make use of LOCK to implement this. In the activate function, where we call the ChangeStockValue method we implement lock. This lock will help in holding the threads from accessing the ChangeStockValue method. This helps in updating the correct value of the stock while we print it on the console and while we save it to the file. This is because unless the event is invoked, the ChangeStockValue method is still in execution. This means that the lock is still on the hold as the method inside the lock is still executing.

Code Snippet:

```
public void Activate()
{
  for (int i = 0; i < 10; i++)
  {
    lock (_object)
    {
      if(Program.detaliedLogOutput)
          Console.WriteLine("Lock Started for Stock : {0}, Initial Value : {1}", this.Name, this.InitialValue);
      ChangeStockValue();
      Thread.Sleep(500);
      if(Program.detaliedLogOutput)
          Console.WriteLine("Lock Released for Stock : {0}, Initial Value : {1}", this.Name, this.InitialValue);
    }
  }
}
```

Sample Output:



```
Lock Started for Stock : Retail, Initial Value : 30
Threshold not reached for Stock : Retail, Initial Value : 30
Lock Released for Stock : Retail, Initial Value : 30
Lock Started for Stock : Banking, Initial Value : 90
Threshold reached for Stock : Banking, Initial Value : 90
Broker 2        Banking         102             5
Broker 3        Banking         102             5
Broker 4        Banking         102             5
Lock Released for Stock : Banking, Initial Value : 90
```