Project 4

William Crutchfield

IntelliJ Ultimate 2017.1.3

Windows 10

To run this Project, simply unzip and import the main project directory “dukeetf” into your IDE. Then, ensure you have the server configured to deploy the “dukeetf:war exploded” artifact at launch. Also, ensure “project4input.txt” is located on your Desktop.

Contents

Changelog

1. DukeETFServlet.java
2. PriceVolumeBean.java
3. main.xhtml

DukeETFServlet.java

Left file: Original DukeETFServlet.java

Right file: Modified DukeETFServlet.java

```
42          public void send(double price, int volume) {
    42      public void send(String date, String time, String price, String volume, String high,
            String low) {
```

```
46          String msg = String.format("%.2f / %d", price, volume);
    46      String msg = String.format("%s , %s , %s , %s , %s , %s", date, time, price, volume,
            high, low); // "%.2f / %d"
```

PriceVolumeBean.java

Left file: Original PriceVolumeBean.java

Right file: Modified PriceVolumeBean.java

```
10          import java.util.Random;
    11      import java.io.BufferedReader;
    12      import java.io.FileNotFoundException;
    13      import java.io.FileReader;
    14      import java.io.IOException;
```

```
27          private Random random;
29          private volatile double price = 100.0;
30          private volatile int volume = 300000;
    32      private volatile String[] data;
    33      private String home = System.getProperty("user.home");  // gets users home folder
            for creating file
    34      private BufferedReader br;
    35      private volatile String date;
    36      private volatile String time;
    37      private volatile String price;
    38      private volatile String volume;
    39      private volatile String low;
    40      private volatile String high;
```

```
37          random = new Random();
39          tservice.createIntervalTimer(1000, 1000, new TimerConfig());
    48      tservice.createIntervalTimer(0, 1000, new TimerConfig());
```

```
55      // Setup BufferedReader to read the project4input.txt file from the users Desktop
56      try {
57        br = new BufferedReader(new FileReader(home + "\\Desktop\\project4input.txt"));
58      } catch (FileNotFoundException e) {
59        e.printStackTrace();
60      }
```

```
48      public void timeout() {
64      public void timeout() throws IOException {
```

```
50      price += 1.0*(random.nextInt(100)-50)/100.0;
51      volume += random.nextInt(5000) - 2500;
52      if (servlet != null)
53      servlet.send(price, volume);
66      data = br.readLine().split(",");
67      date = data[0];
68      time = data[1];
69      price = data[2];
70      volume = data[3];
71      high = data[4];
72      low = data[5];
73      if (servlet != null) {
74        servlet.send(date, time, price, volume, high, low);
75      }
```

main.xhtml

Left file: Original main.xhtml

Right file: Modified main.xhtml

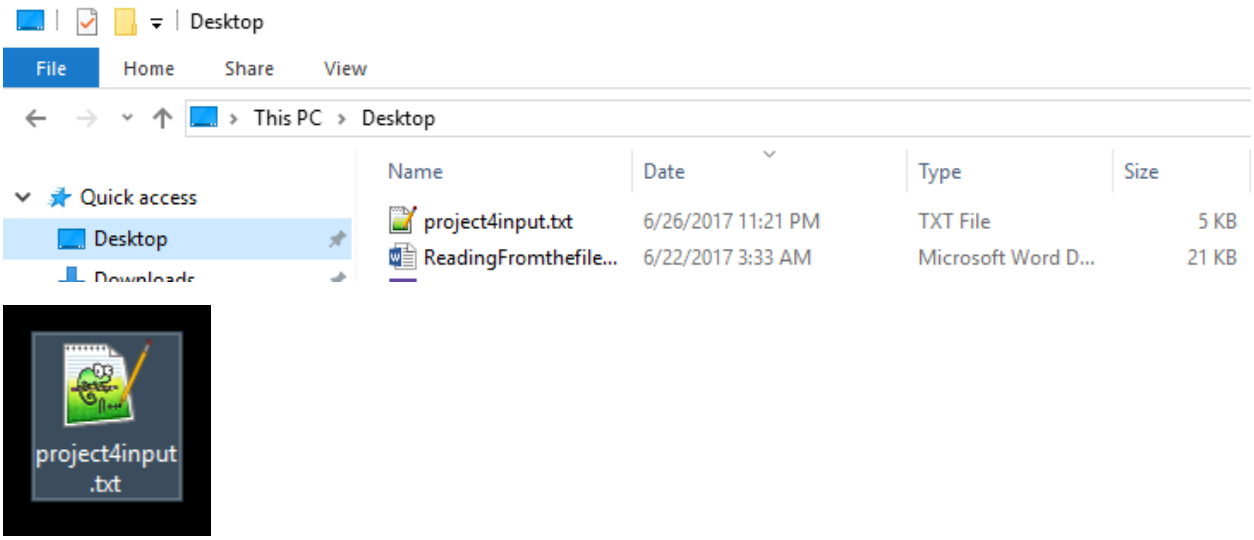
```
21          var arraypv = ajaxRequest.responseText.split("/");
22          document.getElementById("price").innerHTML = arraypv[0];
23          document.getElementById("volume").innerHTML = arraypv[1];
    21      var arraypv = ajaxRequest.responseText.split(",");
    22      document.getElementById("date").innerHTML = arraypv[0];
    23      document.getElementById("time").innerHTML = arraypv[1];
    24      document.getElementById("price").innerHTML = arraypv[2];
    25      document.getElementById("volume").innerHTML = arraypv[3];
    26      document.getElementById("high").innerHTML = arraypv[4];
    27      document.getElementById("low").innerHTML = arraypv[5];
```

```
37          <table>
38          <tr>
39            <td width="100px">Ticker</td>
40            <td align="center">Price</td>
41            <td id="price" style="font-size:24pt;font-weight:bold;">--.--</td>
42          </tr>
43          <tr>
44            <td style="font-size:18pt;font-weight:bold;" width="100px">DKEJ</td>
45            <td align="center">Volume</td>
46            <td id="volume" align="right">--</td>
47          </tr>
48          </table>
```

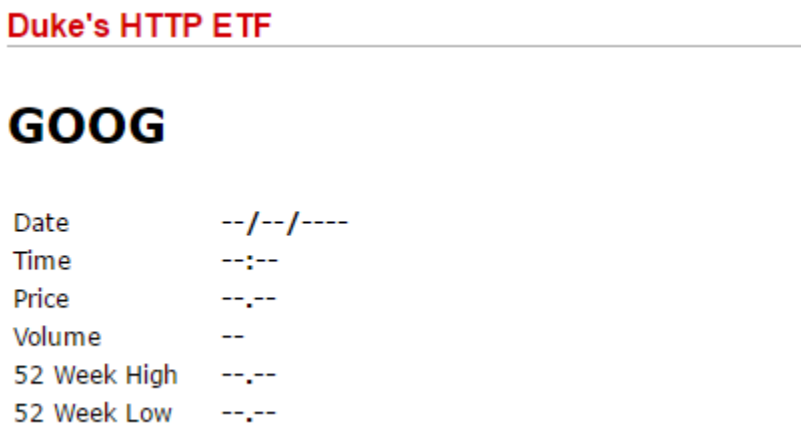
```
41      <p style="font-weight: bold; font-size: 24px"> GOOG </p>
42      <table>
43      <tr>
44        <td align="left" width="100">Date</td>
45        <td id="date" style="font-weight:bold;">--/--/----</td>
46      </tr>
47      <tr>
48        <td align="left" width="100">Time</td>
49        <td id="time" style="font-weight:bold;">--:--</td>
50      </tr>
51      <tr>
52        <td align="left" width="100">Price</td>
53        <td id="price" style="font-weight:bold;">--.--</td>
54      </tr>
55      <tr>
56        <td align="left" width="100">Volume</td>
57        <td id="volume" style="font-weight:bold;">--</td>
58      </tr>
59      <tr>
60        <td align="left" width="100">52 Week High</td>
61        <td id="high" style="font-weight:bold;">--.--</td>
62      </tr>
63      <tr>
64        <td align="left" width="100">52 Week Low</td>
65        <td id="low" style="font-weight:bold;">--.--</td>
66      </tr>
67      </table>
```

Final Product

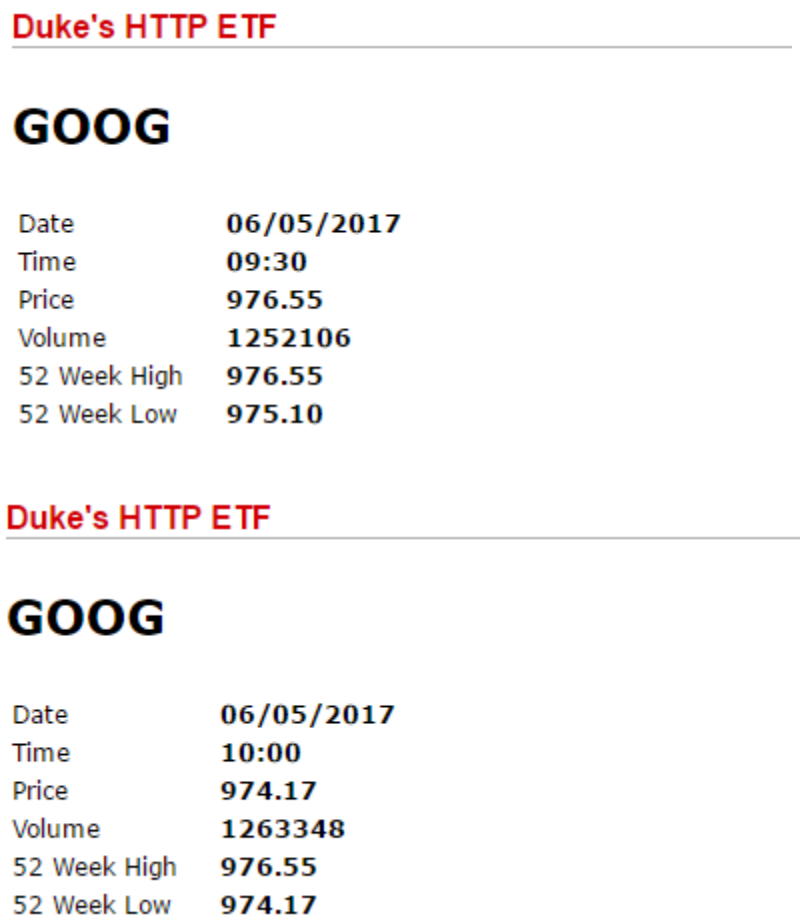
First, before launching the server, we must ensure that the “project4input.txt” file is on the Desktop!



Once that is done, we can the launch the server. We should see a window that looks like this upon running.



Lastly, the server will start reading through the text file, displaying the various information as so.



It will continue to read new lines until reaching the end of the file.

Lessons Learned

I thought this project was a nice challenge! Upon starting this project I was unsure of how to approach the task of reading a new data entry every second. After studying the tutorial project, I was able to determine that the values were updated every second through the “timeout()” method in the “PriceVolumeBean.java” file. I then implemented a BufferedReader to read a new line in the text file each time this method was called, assigning each element in the line to a different variable. The only thing I had trouble with was determining how to point the BufferedReader at the text file on my computer. I eventually found a way to do this by determining where the user’s Desktop is, and reading the file from there. This allows it to work on virtually anyone’s computer, if they put the text file on the Desktop before running the server. Overall, I learned a lot about working with how to read files with java, and using the data gathered to be used in some way. If there is a better way to implement the BufferedReader, please let me know!