Prüfung 3EHIF

daily_values.csv

```
03.01.2000;117,37
04.01.2000;113,56
05.01.2000;111,12
06.01.2000;112,19
07.01.2000;108,63
```

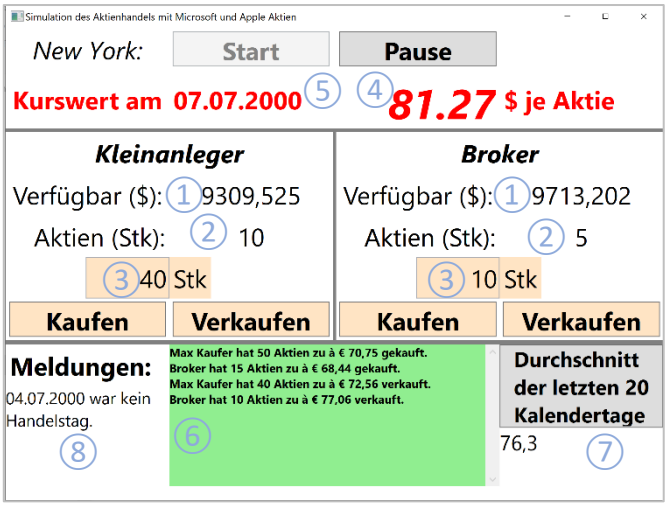
Ein Programm soll den Kursverlauf einer Aktie an der New Yorker Börse simulieren und ermöglichen, dass Aktien gekauft und wieder verkauft werden.

Die Kurswerte sind im File: "StockMarket\data\daily_values.csv".
Sie werden in der Klasse `StockMarketNewYork` eingelesen und gespeichert.

Die Simulationsschleife ist in der Klasse `Simulation`, die eine Instanz von `StockMarketNewYork` speichert. In der Simulationsschleife wird der Aktienwert ④ für das aktuelle Datum `currentDate` ⑤ von `StockMarketNewYork` geholt.

Es gibt zwei Anleger:

- den Kleinanleger Max Kaufer und
- einen professioneller Börsenbroker.



Folgende Klassenhierarchie ist zu implementieren:

Jeder Investor hat die Eigenschaften und Methoden:

- Name
- Verfügbare $ ①
- Aktienbesitz in Stück ②
- Kaufen($jeAktie, Anzahl der gekauften Aktien)
  Kosten = $jeAktie*Anzahl ③ **+ Transaktionskosten** (Verringere die verfügbaren$ des Käufers.)
- Verkaufen($jeAktie, Anzahl der verkauften Aktien)
  Kosten = $jeAktie*Anzahl ③ (Erhöhe die verfügbaren$ des Käufers.)

Kleinanleger: **Transaktionskosten = 1 % von ($jeAktie*Anzahl) + $20**
Broker: **Transaktionskosten = 3 % von ($jeAktie*Anzahl)**
Implementiere die Berechnung der Transaktionskosten als abstract- oder virtual-Methode.

<u>Vervollständige das Simulationsprogramm, so dass es flüssig abläuft und beide Anleger während des Ablaufs, Aktienkäufe und -verkäufe durchführen können</u>

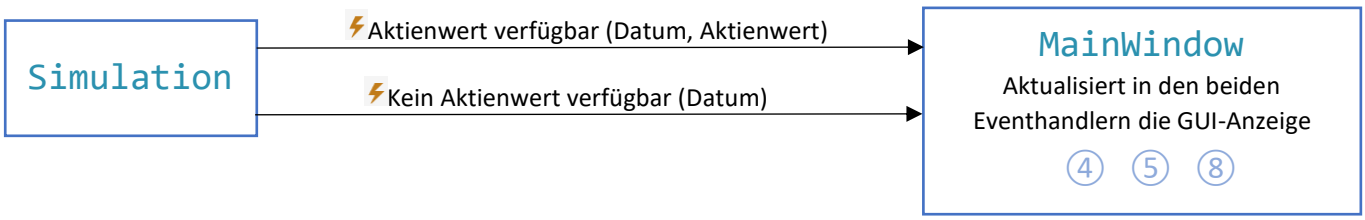
Empfohlene Vorgangsweise: Öffne die Aufgabenansicht in Visual Studio um alle TODOs zu sehen.

1. Mach dich zunächst mit dem Programmcode vertraut. Die drei Investorenklassen sind schon angelegt. Nach dem Starten der Simulation ist das Programm gefreezet, weil noch wichtige Programmteile fehlen.

2. Einbau der Klassenhierarchie: Implementiere die Klassenhierarchie der Investoren. `// TODO 1 und 2`

3. Vervollständige das Fileeinlesen: Speichere das Datum und den dazugehörigen Aktienwert in einer Collectionklasse deiner Wahl. Und vervollständige die `StockMarketNewYork` Klasse. `// TODOs 3 bis 7`

4. Vervollständige die Simulationsschleife `// TODO 9 und 10`
Baue dabei die beiden Events für **`NeuerAktienwertVerfügbar`** `// TODO Event 1` und **`KeinAktienwertGefunden`** ein `// TODO Event 2`.



Prüfung 3EHIF

5. Implementiere den Kauf- und Verkauf von Aktien `// TODO 11 und 12`.
Rufe die Kauf- und Verkaufmethode beim Kleinanleger bzw. dem Broker auf.
Gib Meldungen im TextBox-Control `output` ⑥ aus und aktualisiere ① und ②.

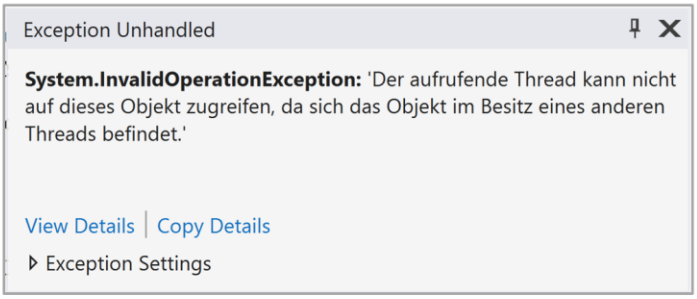
6. Implementiere den Start- und Pause-Button `// TODO 13`.

7. Implementiere die Durchschnittsberechnung der letzten 20 Tage ⑦ `// TODO 14 und 8`.

Hinweis zum Dispatchen:

`Output` ist ein Control, das im GUI-Thread angelegt wurde.

Daher kann ein anderer Thread nicht einfach darauf zugreifen!



Bugfix: Dispatchen Entweder über das Control:

```
Output.Dispatcher.BeginInvoke(
    (Action)(() => { Output.Text += $"{folder}: {size} Bytes\n"; })
);

Output.Dispatcher.Invoke(
    () => { Output.Text += $"{folder}: {size} Bytes\n"; }
);
```

Oder über das MainWindow:

```
Dispatcher.Invoke(
    () => { Output.Text += $"{folder}: {size} Bytes\n"; }
);
```

Viel Erfolg!