Unit Testing mit xUnit in C#

BBZW, Modul 426

Patrick Bucher

28.04.2024

Was passiert in dieser Klasse? Wie soll man diese testen?

```java
public class RuleOfThree {
    private double totalPrice;
    private double amount;
    public RuleOfThree(double totalPrice, double amount) {
        this.totalPrice = totalPrice;
        this.amount = amount;
    }
    public double CalculateFor(double wantedAmount) {
        double singleUnitPrice = this.totalPrice / this.amount;
        return singleUnitPrice * wantedAmount;
    }
}
```

Wir schreiben ein Demo-Programm!

```
public class RuleOfThreeDemo {
    public static void Main(string[] args) {
        RuleOfThree potatoRule = new RuleOfThree(3.50, 12);
        var potatoPrice20 = potatoRule.CalculateFor(20);
        Console.WriteLine($"20 Potatoes for {potatoPrice20:0.00}");

        RuleOfThree bananaRule = new RuleOfThree(2.90, 6);
        var bananaPrice10 = bananaRule.CalculateFor(10);
        Console.WriteLine($"10 Bananas for {bananaPrice10:0.00}");
    }
}
```

Und wie überprüfen wir das jetzt?

```
$ dotnet run --project RuleOfThree.Demo/
```

Ausgabe:

```
20 Potatoes for 5.83
10 Bananas for 4.83
```

Kartoffeln:

$$\frac{3.50}{12} \times 20 = 5.8\overline{3}$$

Bananen:

$$\frac{2.90}{6} \times 10 = 4.8\overline{3}$$

Ergebnis: *korrekt*!

Doch könnte das nicht der Computer für uns rechnen?

```
public class RuleOfThreeDemo {
    public static void Main(string[] args) {
        RuleOfThree potatoRule = new RuleOfThree(3.50, 12);
        var potatoPrice20 = potatoRule.CalculateFor(20);
        Console.WriteLine($"20 Potatoes for {potatoPrice20:0.00}");
        Console.WriteLine("correct? " + (potatoPrice20 == 3.50 / 12 * 20));

        RuleOfThree bananaRule = new RuleOfThree(2.90, 6);
        var bananaPrice10 = bananaRule.CalculateFor(10);
        Console.WriteLine($"10 Bananas for {bananaPrice10:0.00}");
        Console.WriteLine("correct? " + (bananaPrice10 == 2.90 / 6 * 10));
    }
}
```

Hat das funktioniert?

```
$ dotnet run --project RuleOfThree.Demo/
```

Ausgabe:

```
20 Potatoes for 5.83
correct? True
10 Bananas for 4.83
correct? True
```

Ergebnis: *korrekt*!

Doch leider ist die Auswertung etwas umständlich…

Gibt es keine komfortablere Variante?

Doch, z.B. mit xUnit

Einfügen: `using Xunit;`

Assertions verwenden:

- `Assert.Equal(5, 2 + 3);`
- `Assert.NotEqual(5, 3 + 4);`
- `Assert.True(1 < 2);`
- `Assert.False(5 > 10);`
- `Assert.InRange(5, 1, 10);`
- `Assert.NotInRange(15, 1, 10);`

xUnit-Beispiel für `RuleOfThree`

```
public class RuleOfThreeTest {
    [Fact]
    public void TenForTwoIsFiveForOne() {
        // Arrange
        double total = 10.0;
        double amount = 2.0;
        RuleOfThree rule = new RuleOfThree(total, amount);
        // Act
        double actualTotal = rule.CalculateFor(1);
        // Assert
        Assert.Equal(5.0, actualTotal);
    }
}
```

Ausgabe von Testlauf

```
$ dotnet test
Passed!  - Failed:     0, Passed:     1, Skipped:     0, Total:     1, Duration: 5 ms
```

Entwicklungsumgebung

Extension für *Visual Studio Code*:

- C# Dev Kit

Visual Studio 2022 (oder neuer) sollte entsprechende Funktionalität bereits integriert haben.

Auftrag (Einzelarbeit, ca. 60 min.)

1. [10 min.] Mache Sie sich mit dem Beispielcode vertraut.
   - Repository `rule-of-three`
2. [50 min.] Lösen Sie die Übungen.
   - Repository `unittest-csharp`