ToiletPaper #149

Testing with Mockito

Author: Alexander Pöhlmann / Software Engineer / Office Leipzig

Problem

With its possibilities to mock classes, Mockito is a very powerful tool. But when the *equals()* methods of the objects to be checked do not exist or are faulty, Mockito reaches its limits with its *verify()* methods. Normally, the classes are simply fixed and that's it. In the case of external classes, it's not that simple. There are several possibilities: transferring them into their own classes, hiding them very well in a wrapper or switching straight to Kotlin.

Solution

A useful solution to completely test the code is to use *ArgumentCapture*. This makes it possible to capture the instances during *verify()* and check them later.

Example

```
FuelPortion expectedFuelPortion = FuelPortion.DYNAMIC;

Engine myMockEngine = mock(Engine.class); //Or with @Mock
Car myTestCar = new Car(myMockEngine);

myTestCar.startEngine();

ArgumentCaptor<EngineConfiguration> engineConfigurationCapture = ArgumentCaptor.forClass(EngineConfiguration.class); //Or with @Captor
verify(myMockEngine).start(engineConfigurationCapture.capture()); // capture() is the central call for ArgumentCapture

EngineConfiguration actuelEngineConfiguration = engineConfigurationCapture.getValue();
Assert.assertEquals(expectedFuelPortion, actuelEngineConfiguration.getFuelPortion());
```

Further Aspects

- https://www.vogella.com/tutorials/Mockito/article.html
- https://www.baeldung.com/mockito-series
- https://github.com/mockito/mockito