Test-Frameworks

JUnit, Spock, Assertions, Mocks und Best Practices

Was ist das?

- Test-Frameworks ermöglichen das Schreiben und die Ausführung von Tests.
- Test-Frameworks bieten die Möglichkeit Testfälle zu gruppieren und Testergebnisse zu validieren (Assertions).
- Sie stellen die Testergebnisse als Reports zur Verfügung.

Testframeworks Java / Groovy-Umfeld

- TestNG
- JUnit
- Spock

JUnit - http://junit.org

- “JUnit was born on a flight from Zurich to the 1997 OOPSLA in Atlanta.”
- Kent Beck und Erich Gamma entwickelten die erste Version
- Martin Fowler lieferte Feedback
  - “assertion messages are the first argument instead of following the java convention of putting optional arguments at the end....”
- Rot/Grün Balken -> Errungenschaft von JUnit

Test-Zyklus

```
@BeforeClass
static void onyOnce() {...}

@Before
void setup() {...}

@Test
void testName() {...}

@After
void tearDown() {...}

@AfterClass
static void onlyOnce() {...}
```

Annotation / Asserts

`@Ignore` - Test abschalten

```
assertEquals(message, expected, actual)
assertTrue(...)
assertNotNull(...)
assertNull(...)
assertSame(...)
fail()
```

z. B.:

```
String name = “Peter”
assertEquals(“The name is wrong.”, “Peter”, name)
...
```

Beispiel

```
public class FooTest {

    @Test
    public void thisAlwaysPasses() {
    }

    @Test
    @Ignore
    public void thisIsIgnored() {
    }
}
```

Assertion-Framework

- Fest Fluent Assertions 1.4 http://fest.easytesting.org / https://code.google.com/p/fest

Maven3

```
<dependency>
    <groupId>org.easytesting</groupId>
    <artifactId>fest-assert</artifactId>
    <version>1.4</version>
    <scope>test</scope>
</dependency>
```

Gradle - `testCompile "org.easytesting:fest-assert:1.4"`

Fest-Beispiel

```
String s = "Software";
assertThat(s).startsWith("So").endsWith("re");


File f = new File(".....");
File expected = new File(".....");
assertThat(f).exists().isFile().hasSameContentAs
(expected);


List<String> words = Arrays.asList("one", "two");
assertThat(words).contains("one").doesNotHaveDuplicates()
     .containsExactly("one", "two").hasSize(2).
isNotEmpty();
```

Mocks

- Dummy - Attribute oder Methoden Parameter
- Fake - funktionierenden Implementierung mit Abkürzung, nicht für Prod (z.B. In-Memory-Test-DB)
- Stubs - liefern vorgerfertigte Antworten zu Aufrufen
- Spies - sind Stubs die Informationen aufzeichnen je nach Aufruf (z.B. Email-Dienst der Anzahl der gesendeten Nachrichten aufzeichnet)
- Mocks - vorkonfigurierte Erwartungen zu Aufrufen

Siehe auch: http://xunitpatterns.com/Mocks,%20Fakes,%20Stubs%20and%20Dummies.html

Quelle: http://martinfowler.com/bliki/TestDouble.html

Mock Frameworks

- EasyMock
- PowerMock

...

- Mockito
- Spock

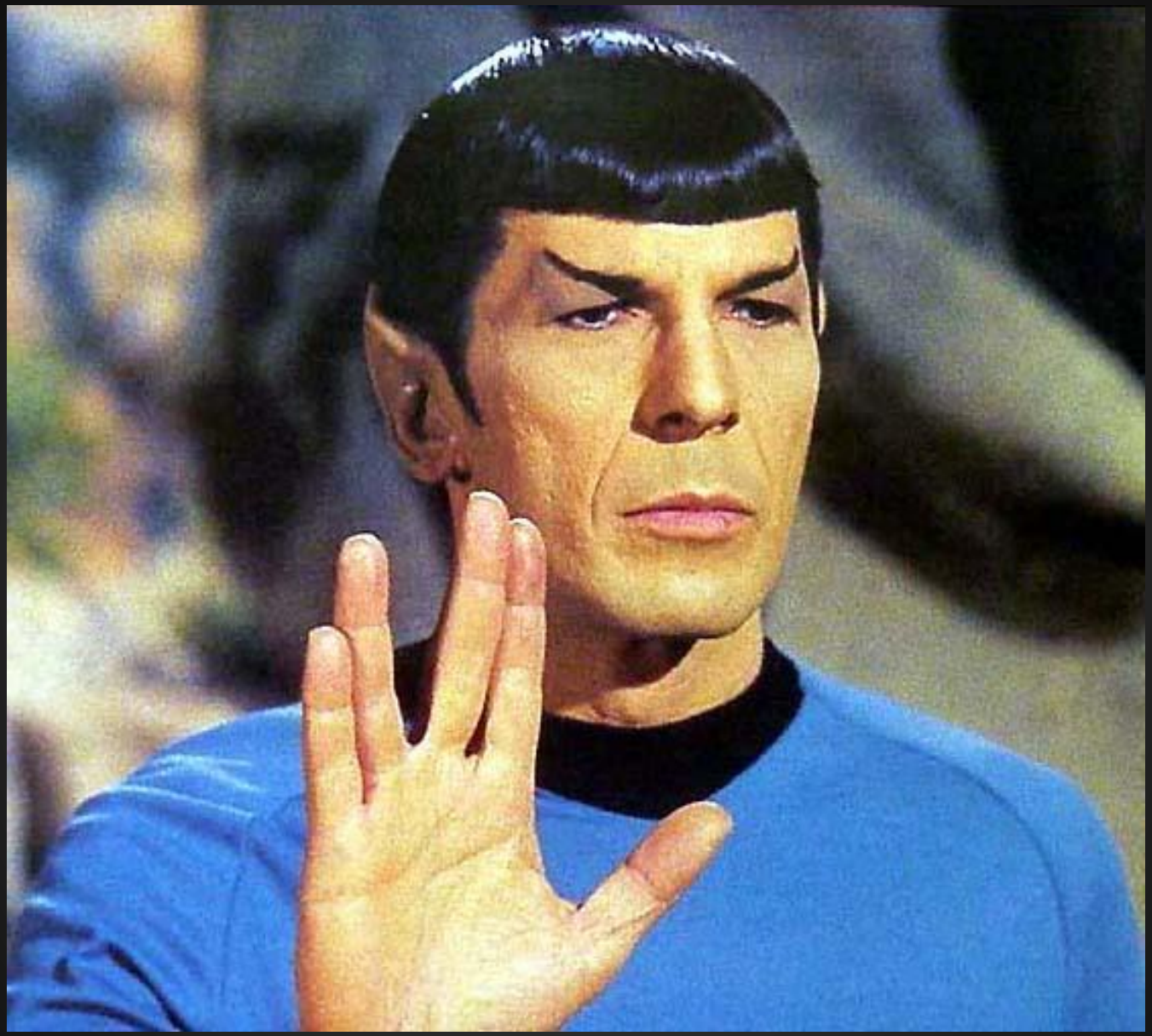
Mocks in den Testebenen

UnitTest - Ebene

- durch Mock-Frameworks

ComponentTest-Ebene

- Stubs und Spies
- z. B. als Controller für Rest-Request
- werden während Tests im Container hochgefahren
- DB (in Memory)



http://www.paperdroids.com/wp-content/uploads/2013/02/spock-leornard-nimoy-star-trek-tos.jpg

Spock

- Spock - the enterprise ready specification framework - https://code.google.com/p/spock
- 05.03.2009 - Spock Framework 0.1 released
- GitHub - https://github.com/spockframework/spock
- starkes assert - Ausagekräftige Meldungen
- given / when / then
- Spock verwendet Groovy

Spock - Methoden

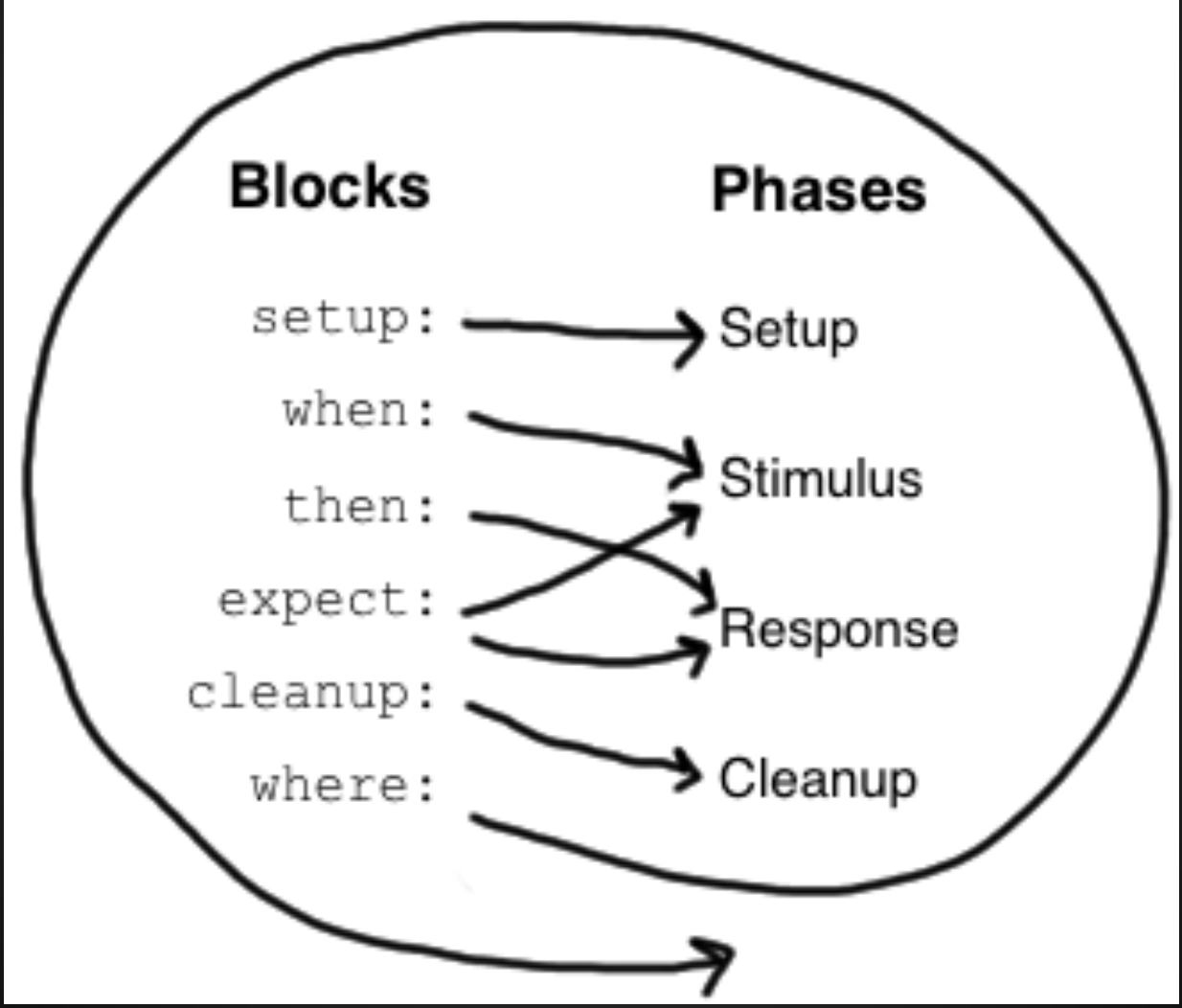
Life-Cycle-Methoden

```
def setup() {}          // läuft vor jeder Test-Methode
def cleanup() {}        // läuft nach jeder Test-Methode
def setupSpec() {}     // läuft einmal pro Klasse
def cleanupSpec() {}   // läuft nach der letzten Methode
```

Feature-Methode (Test-Methode / Testfall)

```
def "pushing an element on the stack"() {
  // blocks go here
}
```

Spock Blocks



Ein Beispiel

```
def "HashMap accepts null key"() {
  setup:
  def map = new HashMap()

  when:
  map.put(null, "elem")

  then:
  notThrown(NullPointerException)
}
```

Where-Blöcke

```
def "computing the maximum of two numbers"() {
  expect:
  Math.max(a, b) == c

  where:
  a << [5, 3]
  b << [1, 9]
  c << [5, 9]
}
```

Spock - BDD Style

```
given: "an empty bank account"
// ...

when: "the account is credited $10"
// ...

then: "the account's balance is $10"
// ...
```

Spock - Interactions

```
def "events are published to all subscribers"() {
  def subscriber1 = Mock(Subscriber)
  def subscriber2 = Mock(Subscriber)
  def publisher = new Publisher()
  publisher.add(subscriber1)
  publisher.add(subscriber2)

  when:
  publisher.fire("event")

  then:
  1 * subscriber1.receive("event")
  1 * subscriber2.receive("event")
}
```

Spock - Interactions

```
def "Controller responds 200 OK when called correctly."()
{
  given:
  def serviceMock = Mock(Service)
  def controller = new Controller(service: serviceMock)

  when:
  def response = controller.index("someValue")

  then:
  1 * serviceMock.read("someValue") >> new Value()
  response != null
  response.status == 200
}
```

Spock - Eine Frage des Stils

```
def serviceMock = Mock(Service)
def controller = new Controller(service: serviceMock)
...

def "Controller responds 200 OK when called correctly."()
{
  given: serviceMock.read("someValue") >> new Value()
  when: def response = controller.index("someValue")
  then: response.status == 200
}
...
```

Spock Assert

Test

```
setup:
def stack = new Stack()
def elem = "push me"

when:
stack.push(elem)

then:
!stack.empty
stack.size() == 1
stack.peek() == elem
```

Ausgabe

```
Condition not satisfied:

assert stack.size() == 1
       |       |    |
       |       2    false
       ...
```

Verwendung

Mit Spock können aus Groovy-Tests:

- Java-Klassen getestet werden
- Groovy-Klassen getestet werden

Spock kann für alle Test-Ebenen eingesetzt werden.

Warum ist Spock eine gute Idee?

- starkes Assert (keine extra Assertion-Bibliothek nötig)
- bringt eigenes Mock-Framwork mit (kein extra Mock-Framework mötig)
- Groovy spart ca. 10% bis 40% Quelltext im Vergleich zu Java
  - Test-Code oft das 2 bis 3 fache als Produktions-Code
  - Einsatz von Groovy für die Tests kann bis zu 30% Code sparen (!)

Benennung von Testfällen

```
void testIndexCalled() {
  given: serviceMock.read("someValue") >> new Value()
  when: def response = controller.index("someValue")
  then: response.status == 200
}

void controllerResponds200OkWhenIndexIsCalled() {
  ...
}

void "Controller responds 200 OK when index is called"() {
  ...
}
```

Benennung von Testfällen II

- Der Name des Test soll wieder geben WAS passiert nicht WIE es passiert.
- Im Namen des Tests soll sich der fachliche UseCase widerspiegeln und nicht die technische Realisierung.
- Das Ergebnis sollte zuerst genannt werden und dann die Bedingung unter der das Ergebnis zustande kommt.

Global Test State

Global State normalerweise in nicht sehr gut. Kann zu Nebenläfigkeitsproblemen führen (je nach Thread-Modell der Anwendung).

- Verhindert das aufblähen von Methoden Signaturen
- Führt zu wartbaren und verständlicheren Code
- Vermeidet das Problem der Rückgabewerte

Beispiel zu Global Test State

```
def serviceMock = Mock(Service)
def controller = new Controller(service: serviceMock)
...

def "Controller responds 200 OK when called correctly."()
{
  given: serviceMock.read("someValue") >> new Value()
  when: def response = controller.index("someValue")
  then: response.status == 200
}
...
```

Beispiel zu Global Test State

```
def serviceMock = Mock(Service)
def controller = new Controller(service: serviceMock)
...
def response
...
def "Controller responds 200 OK when called correctly."()
{
  given: serviceMock.read("someValue") >> new Value()
  when: response = controller.index("someValue")
  then: responseIs 200
}

private void responseIs(code) {
  response.status == code
}
```