Star of Java, 27.07.2023

Testing Tools in Java

Sandra Parsick

@SandraParsick
@sparsick@mastodon.social
mail@sandra-parsick.de

About me

- Sandra Parsick
- Freelance Software Developer
- Topics:
  - Java Enterprise
  - Agile methods
  - Software Craftmanship
  - Automation of Development Process
- Trainings
- Workshops

mail@sandra-parsick.de

@SandraParsick

@sparsick@mastodon.social

https://www.sandra-parsick.de

https://ready-for-review.dev


This cannot be
not be tested locally.
This is too
time-consuming
to test.
You can’t test it.


CHALLENGE ACCEPTED
imgflip.com
quickmeme.com

Agenda

Integrated tests

equals, hashCode, toString

Test Code Duplication

Test Data

Concurrency

Poorly readable assertions

Tests that run only under certain conditions

Problem: Tests that run only under certain conditions

JUnit 5 – Conditional Test Execution

```
3     //  Operating System Conditions
4
5     @Test
6     @EnabledOnOs(OS.LINUX)
7     void testForLinux(){
8         assertTrue(true);
9     }
10
11    @Test
12    @EnabledOnOs(OS.WINDOWS)
13    void testForWindows(){
14        assertTrue(true);
15    }
16
17    @Test
18    @EnabledOnOs({OS.LINUX, OS.WINDOWS})
19    void testForLinuxAndWindows(){
20        assertTrue(true);
21    }
22
23    @Test
24    @DisabledOnOs(OS.LINUX)
25    void testNotForLinux(){
26        assertTrue(true);
27    }
```

```
29    // Java Runtime Environment Conditions
30
31    @Test
32    @EnabledOnJre(JRE.JAVA_8)
33    void onlyOnJava8() {
34        assertTrue(true);
35    }
36
37    @Test
38    @EnabledOnJre({ JRE.JAVA_9, JRE.JAVA_10 })
39    void onJava9Or10() {
40        assertTrue(true);
41    }
42
43    @Test
44    @DisabledOnJre(JRE.JAVA_8)
45    void notOnJava8() {
46        assertTrue(true);
47    }
```

JUnit 5 – Conditional Test Execution

```
50      // System Property Conditions
51
52      @Test
53      @EnabledIfSystemProperty(named = "os.arch", matches = ".*64.*")
54      void onlyOn64BitArchitectures() {
55          assertTrue(true);
56
57      }
58
59      @BeforeEach
60      void setup(){
61          System.setProperty("ci-server", "true");
62      }
63
64      @Test
65      @DisabledIfSystemProperty(named = "ci-server", matches = "true")
66      @EnabledIfSystemProperty(named = "ci-server", matches = "true")
67      void notOnCiServer() {
68          assertTrue(true);
69      }
```

JUnit 5 – Conditional Test Execution

```
71      // Environment Variable Conditions
72      // variable can be set in idea run configuration
73      // ENV1=staging-server
74      // ENV2=local.development
75
76      @Test
77      @EnabledIfEnvironmentVariable(named = "ENV1", matches = "staging-server")
78      void onlyOnStagingServer() {
79          assertTrue(true);
80      }
81
82      @Test
83      @DisabledIfEnvironmentVariable(named = "ENV2", matches = ".*development.*")
84      void notOnDeveloperWorkstation() {
85          assertTrue(true);
86      }
87
```

Problem: Test Code Duplication

```java
public class Position {

    private int xCoordinate;
    private int yCoordinate;

    public Position(int xCoordinate, int yCoordinate) {
        this.xCoordinate = xCoordinate;
        this.yCoordinate = yCoordinate;
    }

    public Position moveBackward(Direction facingDirection) {
        int newX = xCoordinate;
        int newY = yCoordinate;
        switch (facingDirection) {
            case NORTH:
                newY--;
                break;
            case SOUTH:
                newY++;
                break;
            case WEST:
                newX++;
                break;
            case EAST:
                newX--;
                break;
        }
        return new Position (newX, newY);
    }

    public Position moveForward(Direction facingDirection) {
        int newX = xCoordinate;
        int newY = yCoordinate;

        switch (facingDirection) {
            case NORTH:
                newY++;
                break;
            case SOUTH:
                newY--;
                break;
            case WEST:
                newX--;
                break;
            case EAST:
                newX++;
                break;
        }

        return new Position(newX, newY);
    }
}
```

Naive Approach

```
7  class PositionNaiveTest {
8
9      @Test
10     void moveBackward_toNorth() {
11         Position positionUnderTest = new Position(10, 10);
12
13         Position newPosition = positionUnderTest.moveBackward(Direction.NORTH);
14
15         assertThat(newPosition).isEqualTo(new Position(10, 9));
16     }
17
18     @Test
19     void moveBackward_toEast() {
20         Position positionUnderTest = new Position(10, 10);
21
22         Position newPosition = positionUnderTest.moveBackward(Direction.EAST);
23
24         assertThat(newPosition).isEqualTo(new Position(9, 10));
25     }
26
```

```
27      @Test
28      void moveBackward_toSouth() {
29          Position positionUnderTest = new Position(10, 10);
30
31          Position newPosition = positionUnderTest.moveBackward(Direction.SOUTH);
32
33          assertThat(newPosition).isEqualTo(new Position(10, 11));
34      }
35
36      @Test
37      void moveBackward_toWest() {
38          Position positionUnderTest = new Position(10, 10);
39
40          Position newPosition = positionUnderTest.moveBackward(Direction.WEST);
41
42          assertThat(newPosition).isEqualTo(new Position(11, 10));
43      }
44  }
```

Parametrized Tests With JUnit 4

```java
@RunWith(Parameterized.class)
public class PositionMoveBackwardJUnit4Test {

    private final Direction direction;
    private final Position expectedPosition;

    public PositionMoveBackwardJUnit4Test(Direction direction, Position expectedPosition) {
        this.direction = direction;
        this.expectedPosition = expectedPosition;
    }

    @Parameterized.Parameters
    public static Collection moveBackwardParameter() {
        return Arrays.asList(new Object[][]{
                {Direction.NORTH, new Position(10, 9)},
                {Direction.EAST, new Position(9, 10)},
                {Direction.SOUTH, new Position(10, 11)},
                {Direction.WEST, new Position(11, 10)}
        });
    }

    @Test
    public void moveBackward() {
        Position positionUnderTest = new Position(10, 10);
        Position newPosition = positionUnderTest.moveBackward(direction);
        assertThat(newPosition).isEqualTo(expectedPosition);
    }
}
```

```java
@RunWith(Parameterized.class)
public class PositionMoveForwardJUnit4Test {

    private final Direction direction;
    private final Position expectedPosition;

    public PositionMoveForwardJUnit4Test(Direction direction, Position expectedPosition) {
        this.direction = direction;
        this.expectedPosition = expectedPosition;
    }

    @Parameterized.Parameters
    public static Collection moveBackwardParameter() {
        return Arrays.asList(new Object[][]{
                {Direction.NORTH, new Position(10, 11)},
                {Direction.EAST, new Position(11, 10)},
                {Direction.SOUTH, new Position(10, 9)},
                {Direction.WEST, new Position(9, 10)}
        });
    }

    @Test
    public void moveForward() {
        Position positionUnderTest = new Position(10, 10);

        Position newPosition = positionUnderTest.moveForward(direction);

        assertThat(newPosition).isEqualTo(expectedPosition);
    }
}
```

Parametrized Test With JUnit5

```java
class PositionJUnit5Test {

    @ParameterizedTest
    @MethodSource("createMoveBackwardParameter")
    void moveBackward(Direction direction, Position expectedPosition) {
        Position positionUnderTest = new Position(10, 10);
        Position newPosition = positionUnderTest.moveBackward(direction);
        assertThat(newPosition).isEqualTo(expectedPosition);
    }

    private static Stream<Arguments> createMoveBackwardParameter() {
        return Stream.of(
                Arguments.of(Direction.NORTH, new Position(10, 9)),
                Arguments.of(Direction.EAST, new Position(9, 10)),
                Arguments.of(Direction.SOUTH, new Position(10, 11)),
                Arguments.of(Direction.WEST, new Position(11, 10))
        );
    }

    @ParameterizedTest
    @MethodSource("createMoveForwardParameter")
    void moveForward(Direction direction, Position expectedPosition) {
        Position positionUnderTest = new Position(10, 10);
        Position newPosition = positionUnderTest.moveForward(direction);
        assertThat(newPosition).isEqualTo(expectedPosition);
    }

    private static Stream<Arguments> createMoveForwardParameter() {
        return Stream.of(
                Arguments.of(Direction.NORTH, new Position(10, 11)),
                Arguments.of(Direction.EAST, new Position(11, 10)),
                Arguments.of(Direction.SOUTH, new Position(10, 9)),
                Arguments.of(Direction.WEST, new Position(9, 10))
        );
    }
}
```

Further Options in JUnit5

```java
@ParameterizedTest
@ValueSource(strings = {"a", "b", "c"})
void singleLetter(String candidate) {
    assertTrue(isSingleLetter(candidate));
}

private boolean isSingleLetter(String candidate) {
    return candidate.length() == 1;
}

// type for value sources:
//    short
//    byte
//    int
//    long
//    float
//    double
//    char
//    java.lang.String
//    java.lang.Class


@ParameterizedTest
@EnumSource(TimeUnit.class)
void timeUnit(TimeUnit timeUnit) {
    assertThat(timeUnit).isNotNull();
}
```

```java
@ParameterizedTest
@CsvSource({ "foo, 1", "bar, 2",
             "'baz, qux', 3" })
// @CsvFileSource(resources = "/test-data.csv", numLinesToSkip = 1)
void testWithCsvSource(String first, int second) {
    assertNotNull(first);
    assertNotEquals(0, second);
}

@ParameterizedTest
@ArgumentsSource(MyArgumentsProvider.class)
void testWithArgumentsSource(String argument) {
    assertNotNull(argument);
}
```

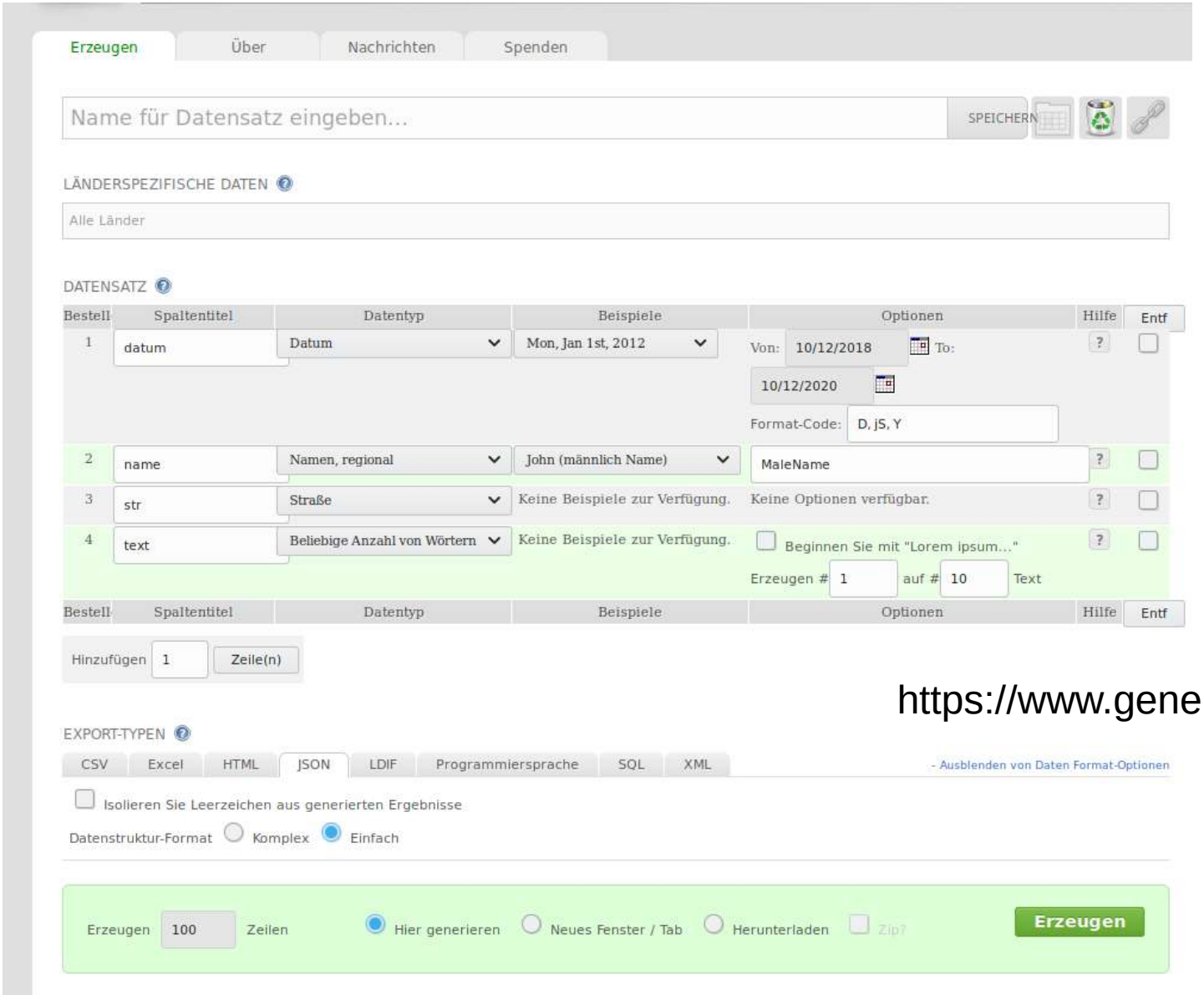
```java
public class MyArgumentsProvider implements ArgumentsProvider {
    @Override
    public Stream<? extends Arguments> provideArguments(
                          ExtensionContext extensionContext) {
        return Stream.of("foo", "bar").map(Arguments::of);
    }
}
```

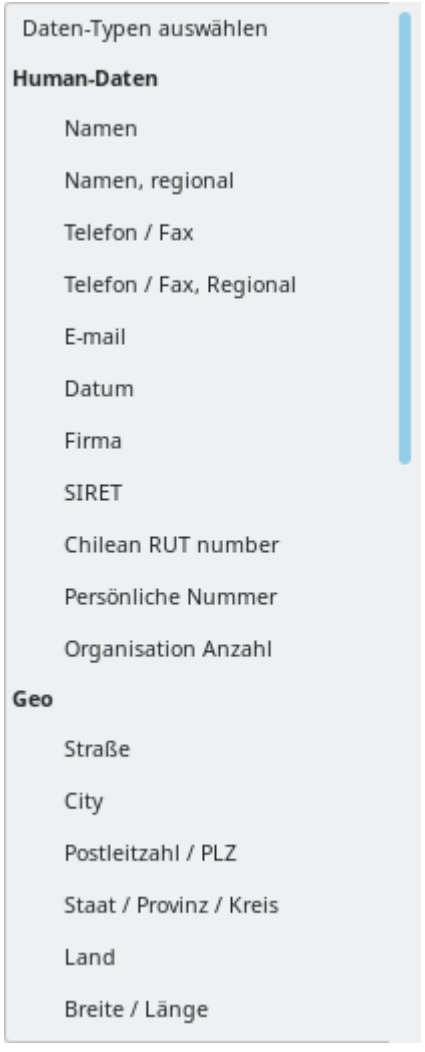
Parametrized Test With Spock

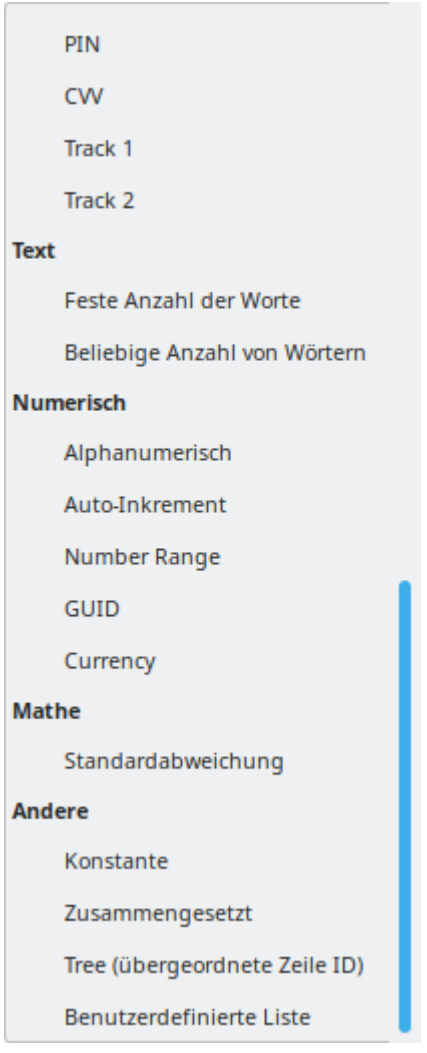
```groovy
class PositionSpockTest extends Specification {

    def "move backward" (Direction direction, Position expectedPosition){
        Position positionUnderTest = new Position(10, 10)
        Position newPosition = positionUnderTest.moveBackward(direction)

        expect:
        assert newPosition == expectedPosition

        where:
        direction | expectedPosition
        Direction.NORTH | new Position(10, 9)
        Direction.EAST | new Position(9, 10)
        Direction.SOUTH | new Position(10, 11)
        Direction.WEST |new Position(11, 10)
    }

    def "move forward" (Direction direction, Position expectedPosition){
        Position positionUnderTest = new Position(10, 10)
        Position newPosition = positionUnderTest.moveForward(direction)

        expect:
        assert newPosition == expectedPosition

        where:
        direction | expectedPosition
        Direction.NORTH | new Position(10, 11)
        Direction.EAST | new Position(11, 10)
        Direction.SOUTH | new Position(10, 9)
        Direction.WEST |new Position(9, 10)
    }
}
```

Problem: Test Data

https://www.generatedata.com


Erzeugen
Über
Nachrichten
Spenden
Name für Datensatz eingeben...
SPEICHERN
LÄNDERSPEZIFISCHE DATEN
Alle Länder
DATENSATZ
Bestell
Spaltentitel
Datentyp
Beispiele
Optionen
Hilfe
Entf
1
datum
Datum
Mon, Jan 1st, 2012
Von: 10/12/2018 To:
10/12/2020
Format-Code: D, jS, Y
2
name
Namen, regional
John (männlich Name)
MaleName
3
str
Straße
Keine Beispiele zur Verfügung.
Keine Optionen verfügbar.
4
text
Beliebige Anzahl von Wörtern
Keine Beispiele zur Verfügung.
Beginnen Sie mit "Lorem ipsum..."
Erzeugen # 1 auf # 10 Text
Hinzufügen 1 Zeile(n)
https://www.generatedata.com
EXPORT-TYPEN
CSV
Excel
HTML
JSON
LDIF
Programmiersprache
SQL
XML
- Ausblenden von Daten Format-Optionen
Isolieren Sie Leerzeichen aus generierten Ergebnisse
Datenstruktur-Format Komplex Einfach
Erzeugen 100 Zeilen
Hier generieren
Neues Fenster / Tab
Herunterladen
Zip?
Erzeugen


Daten-Typen auswählen
Human-Daten
Namen
Namen, regional
Telefon / Fax
Telefon / Fax, Regional
E-mail
Datum
Firma
SIRET
Chilean RUT number
Persönliche Nummer
Organisation Anzahl
Geo
Straße
City
Postleitzahl / PLZ
Staat / Provinz / Kreis
Land
Breite / Länge


PIN
CVV
Track 1
Track 2
Text
Feste Anzahl der Worte
Beliebige Anzahl von Wörtern
Numerisch
Alphanumerisch
Auto-Inkrement
Number Range
GUID
Currency
Mathe
Standardabweichung
Andere
Konstante
Zusammengesetzt
Tree (übergeordnete Zeile ID)
Benutzerdefinierte Liste

Erzeugen | Über | Nachrichten | Spenden

Erstellt **100** **100** Ergebnisse

abbrechen


```
[
    {
        "datum": "Wed, 1st, 2020",
        "name": "Zachary",
        "str": "Ap #213-6985 Molestie  Straße",
        "text": "nec mauris blandit mattis. Cras eget"
    },
    {
        "datum": "Fri, 10th, 2019",
        "name": "Macaulay",
        "str": "Ap #891-1080 Curabitur  Ave",
        "text": "hendrerit neque. In ornare sagittis felis. Donec tempor, est ac"
    },
    {
        "datum": "Sun, 24th, 2019",
        "name": "Jonas",
        "str": "Ap #942-7252 Quam  Avenue",
        "text": "Nam tempor diam dictum sapien. Aenean massa. Integer vitae nibh."
    },
    {
        "datum": "Thu, 4th, 2019",
        "name": "Adrian",
        "str": "6794 Egestas,  Rd.",
        "text": "turpis. Aliquam adipiscing lobortis risus."
    },
    {
        "datum": "Mon, 3rd, 2018",
        "name": "Lev",
        "str": "Ap #105-239 Arcu  Straße",
        "text": "Sed id risus"
    },
```


« Regenerieren

A A A

And what if you prefer code?

Test Data With DataFaker

```java
@Test
void simpleFaker() {
    Faker dataFaker = new Faker();
    Person person = new Person();
    person.setFirstName(dataFaker.name().firstName());
    person.setLastName(dataFaker.name().lastName());
    person.setJobTitle(dataFaker.job().title());

    // more test code
}
```

- Built-In Faker (Extract)
  - Business
  - Commerce
  - Name
  - Lorem
- Local Support (Extract)
  - de (in many variants)
  - en (in many variants)

ObjectMother Pattern

```
class PersonTestData { // ObjectMother pattern
    static Person newPersonWithoutJobTitle() {
        Faker dataFaker = new Faker();
        Person person = new Person();
        person.setFirstName(dataFaker.name().firstName());
        person.setLastName(dataFaker.name().lastName());
        return person;
    }


    static Person newPerson() {
        Faker dataFaker = new Faker();
        Person person = new Person();
        person.setFirstName(dataFaker.name().firstName());
        person.setLastName(dataFaker.name().lastName());
        person.setJobTitle(dataFaker.job().title());
        return person;
    }
}
```

```
@Test
void objectMother(){
    Person personWithoutJobTitle =
              PersonTestData.newPersonWithoutJobTitle();
    Person fullPerson =
              PersonTestData.newPerson();

    //more test code
}
```

TestDataBuilder Pattern

```
class PersonTestDataBuilder { // test data builder pattern
    private String firstName;
    private String lastName;
    private String jobTitle;

    PersonTestDataBuilder withFirstName(String firstName){
        this.firstName = firstName;
        return this;
    }

    PersonTestDataBuilder withLastName(String lastName){
        this.lastName = lastName;
        return this;
    }

    PersonTestDataBuilder withJobTitle (String jobTitle){
        this.jobTitle = jobTitle;
        return this;
    }

    Person build() {
        Person person = new Person();
        person.setFirstName(firstName);
        person.setLastName(lastName);
        person.setJobTitle(jobTitle);
        return person;
    }
}
```

```
@Test
void testDataBuilder(){
    Faker dataFaker = new Faker();
    PersonTestDataBuilder personBuilder = new PersonTestDataBuilder();
    personBuilder.withFirstName(dataFaker.name().firstName())
            .withLastName(dataFaker.name().lastName())
            .withJobTitle(dataFaker.job().title());
    Person person = personBuilder.build();
}
```

Problem: Poorly Readable Assertion

Built-In Assertion in JUnit

```
@Test
void builtInAssertion() {
    Hero hero = new Hero("Batman", "Bruce Wayne");

    assertEquals(hero.getName(), "Batman");
    assertEquals(hero.getRealName(), "Bruce Wayne");
}
```

AssertJ – Fluent Assertion API

```
@Test
void hasField(){
    assertThat(new Hero ("Superman", "Clark Kent"))
      .hasFieldOrPropertyWithValue("realName", "Clark Kent")
      .hasFieldOrPropertyWithValue("name", "Superman");
}
```

```
@Test
void basic() {
    assertThat("The Lord of the Rings").isNotNull()
            .startsWith("The")
            .contains("Lord")
            .endsWith("Rings");
}
```

AssertJ – List Assertion

```java
@Test
void basic(){
    List<String> heros = List.of("Batman", "Superman");

    assertThat(heros)
            .hasSize(2)
            .contains("Batman")
            .containsExactly("Batman", "Superman")
            .containsAnyOf("Batman", "Superman", "Wonder woman");

    // containsNull
    // negotation
}

@Test
void extractionSample(){
    List<Hero> heros = List.of(new Hero("Batman", "Bruce Wayne"), new Hero("Superman", "Clark Kent"));

    assertThat(heros)
            .extracting("realName")
            .contains("Bruce Wayne", "Clark Kent");
}
```

AssertJ – Exception Assertion

```
@Test
void bddStyle(){
    // GIVEN
    String[] names = { "Pier ", "Pol", "Jak" };
    // WHEN
    Throwable thrown = catchThrowable(() -> System.out.println(names[9]));
    // THEN
    assertThat(thrown)
            .isInstanceOf(ArrayIndexOutOfBoundsException.class)
            .hasMessageContaining("9");
}

@Test
void assertThatThrownByExample(){
    assertThatThrownBy(() -> { throw new Exception("boom!"); })
            .isInstanceOf(Exception.class)
            .hasMessageContaining("boom")
            .hasMessage("boom!");
}
```

AssertJ – Exception Assertion

```java
@Test
void assertThatExceptionOfTypeExample(){
    assertThatExceptionOfType(IOException.class)
            .isThrownBy(() -> { throw new IOException("boom!"); })
            .withMessage("%s!", "boom")
            .withMessageContaining("boom")
            .withNoCause();

    //This later syntax has been enriched for common exceptions :
    //    assertThatNullPointerException
    //    assertThatIllegalArgumentException
    //    assertThatIllegalStateException
    //    assertThatIOException
}

@Test
void assertThatNoExceptionIsThrown(){
    assertThatCode(() -> {
        // code that should NOT throw an exception
    }).doesNotThrowAnyException();
}
```

AssertJ - Assumption

```
@Test
void assume(){
    assumeThat("Bonn").isEqualTo("Bonn");
}

@Test
void assumeMoreComplex(){
    assumeThat(new File("/starwars-testdata/star-wars-logo.png"))
            .isFile()
            .exists();
    assumeThat(List.of("Hello", "World"))
            .hasSize(2)
            .contains("Hello");
}
```

JUnit 5 – Grouped Assertion

```
@Test
void standardAssertions() {
    assertEquals(2, 2);
    assertEquals(4, 4, "The optional assertion message is now the last parameter.");
    assertTrue('a' < 'b', () -> "Assertion messages can be lazily evaluated -- "
            + "to avoid constructing complex messages unnecessarily.");
}

@Test
void groupedAssertions() {
    // In a grouped assertion all assertions are executed, and any
    // failures will be reported together.
    assertAll("person",
            () -> assertEquals("John","John"),
            () -> assertEquals("Doe", "Doe"),
            () -> assertThat(Lists.list("foo")).isNotEmpty()
    );
}
```

Exkurs: Migration to JUnit 5

```xml
<properties>
    <junit.jupiter.version>5.4.2</junit.jupiter.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-api</artifactId>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-engine</artifactId>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.junit.vintage</groupId>
        <artifactId>junit-vintage-engine</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>
```

```xml
<dependencyManagement>
    <dependencies>
        <dependency>
            <groupId>org.junit</groupId>
            <artifactId>junit-bom</artifactId>
            <version>${junit.jupiter.version}</version>
            <scope>import</scope>
            <type>pom</type>
        </dependency>
    </dependencies>
</dependencyManagement>
```

Large-Scale Automated Source Code Refactoring

Example: JUnit4 to JUnit5

```
<plugin>
  <groupId>org.openrewrite.maven</groupId>
  <artifactId>rewrite-maven-plugin</artifactId>
  <version>4.38.0</version>
  <configuration>
    <activeRecipes>
      <recipe>org.openrewrite.java.spring.boot2.SpringBoot2JUnit4to5Migration</recipe>
    </activeRecipes>
  </configuration>
  <dependencies>
    <dependency>
      <groupId>org.openrewrite.recipe</groupId>
        <artifactId>rewrite-spring</artifactId>
        <version>4.29.0</version>
    </dependency>
  </dependencies>
</plugin>
```

```
$ mvn rewrite:run
[INFO] Using active recipe(s) [org.openrewrite.java.spring.boot2.SpringBoot2JUnit4to5Migration]
[INFO] Using active styles(s) []
[INFO] Validating active recipes...
[INFO] Project [petclinic] Resolving Poms...
[INFO] Project [petclinic] Parsing Source Files
[INFO] Running recipe(s)...
[WARNING] Changes have been made to pom.xml by:
[WARNING]     org.openrewrite.java.spring.boot2.SpringBoot2JUnit4to5Migration
[WARNING]         org.openrewrite.java.testing.junit5.JUnit4to5Migration
[WARNING]             org.openrewrite.maven.ExcludeDependency: {groupId=junit, artifactId=junit}
[WARNING]             org.openrewrite.maven.ExcludeDependency: {groupId=org.junit.vintage, artifactId=junit-vintage-engine}
[WARNING] Changes have been made to src/test/java/org/springframework/samples/petclinic/owner/OwnerControllerTests.java by:
[WARNING]     org.openrewrite.java.spring.boot2.SpringBoot2JUnit4to5Migration
[WARNING]         org.openrewrite.java.testing.junit5.JUnit4to5Migration
[WARNING]             org.openrewrite.java.testing.junit5.UpdateBeforeAfterAnnotations
[WARNING]             org.openrewrite.java.testing.junit5.UpdateTestAnnotation
[WARNING]         org.openrewrite.java.spring.boot2.UnnecessarySpringRunWith
[WARNING]             org.openrewrite.java.testing.junit5.RunnerToExtension: {runners=
[org.springframework.test.context.junit4.SpringRunner, org.springframework.test.context.junit4.SpringJUnit4ClassRunner],
extension=org.springframework.test.context.junit.jupiter.SpringExtension}
[WARNING]         org.openrewrite.java.spring.boot2.UnnecessarySpringExtension
[WARNING] Changes have been made to src/test/java/org/springframework/samples/petclinic/owner/VisitControllerTests.java by:
[WARNING]     org.openrewrite.java.spring.boot2.SpringBoot2JUnit4to5Migration
[WARNING]         org.openrewrite.java.testing.junit5.JUnit4to5Migration
[WARNING]             org.openrewrite.java.testing.junit5.UpdateBeforeAfterAnnotations
[WARNING]             org.openrewrite.java.testing.junit5.UpdateTestAnnotation
[WARNING]         org.openrewrite.java.spring.boot2.UnnecessarySpringRunWith
[WARNING]             org.openrewrite.java.testing.junit5.RunnerToExtension: {runners=
[org.springframework.test.context.junit4.SpringRunner, org.springframework.test.context.junit4.SpringJUnit4ClassRunner],
extension=org.springframework.test.context.junit.jupiter.SpringExtension}
[WARNING]         org.openrewrite.java.spring.boot2.UnnecessarySpringExtension
```

```
extension=org.springframework.test.context.junit.jupiter.SpringExtension}
[WARNING]         org.openrewrite.java.spring.boot2.UnnecessarySpringExtension
[WARNING] Changes have been made to src/test/java/org/springframework/samples/petclinic/vet/VetControllerTests.java by:
[WARNING]     org.openrewrite.java.spring.boot2.SpringBoot2JUnit4to5Migration
[WARNING]         org.openrewrite.java.testing.junit5.JUnit4to5Migration
[WARNING]             org.openrewrite.java.testing.junit5.UpdateBeforeAfterAnnotations
[WARNING]             org.openrewrite.java.testing.junit5.UpdateTestAnnotation
[WARNING]         org.openrewrite.java.spring.boot2.UnnecessarySpringRunWith
[WARNING]             org.openrewrite.java.testing.junit5.RunnerToExtension: {runners=
[org.springframework.test.context.junit4.SpringRunner, org.springframework.test.context.junit4.SpringJUnit4ClassRunner],
extension=org.springframework.test.context.junit.jupiter.SpringExtension}
[WARNING]         org.openrewrite.java.spring.boot2.UnnecessarySpringExtension
[WARNING] Changes have been made to src/test/java/org/springframework/samples/petclinic/vet/VetTests.java by:
[WARNING]     org.openrewrite.java.spring.boot2.SpringBoot2JUnit4to5Migration
[WARNING]         org.openrewrite.java.testing.junit5.JUnit4to5Migration
[WARNING]             org.openrewrite.java.testing.junit5.UpdateTestAnnotation
[WARNING] Changes have been made to src/test/java/org/springframework/samples/petclinic/service/ClinicServiceTests.java by:
[WARNING]     org.openrewrite.java.spring.boot2.SpringBoot2JUnit4to5Migration
[WARNING]         org.openrewrite.java.testing.junit5.JUnit4to5Migration
[WARNING]             org.openrewrite.java.testing.junit5.UpdateTestAnnotation
[WARNING]         org.openrewrite.java.spring.boot2.UnnecessarySpringRunWith
[WARNING]             org.openrewrite.java.testing.junit5.RunnerToExtension: {runners=
[org.springframework.test.context.junit4.SpringRunner, org.springframework.test.context.junit4.SpringJUnit4ClassRunner],
extension=org.springframework.test.context.junit.jupiter.SpringExtension}
[WARNING]         org.openrewrite.java.spring.boot2.UnnecessarySpringExtension
[WARNING] Please review and commit the results.
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  36.321 s
[INFO] Finished at: 2022-12-13T12:17:55+01:00
[INFO] ------------------------------------------------------------------------
```

```
$ git diff

diff --git a/pom.xml b/pom.xml
index cd0f967..c8db6dc 100644
--- a/pom.xml
+++ b/pom.xml
@@ -62,6 +62,16 @@
       <groupId>org.springframework.boot</groupId>
       <artifactId>spring-boot-starter-test</artifactId>
       <scope>test</scope>
+      <exclusions>
+        <exclusion>
+          <groupId>junit</groupId>
+          <artifactId>junit</artifactId>
+        </exclusion>
+        <exclusion>
+          <groupId>org.junit.vintage</groupId>
+          <artifactId>junit-vintage-engine</artifactId>
+        </exclusion>
+      </exclusions>
     </dependency>

diff --git a/src/test/java/org/springframework/samples/petclinic/PetclinicIntegrationTests.java b/src/test/java/org/springframework
/samples/petclinic/PetclinicIntegrationTests.java
index 669dbdd..1db15a8 100644
--- a/src/test/java/org/springframework/samples/petclinic/PetclinicIntegrationTests.java
+++ b/src/test/java/org/springframework/samples/petclinic/PetclinicIntegrationTests.java
@@ -16,15 +16,12 @@

 package org.springframework.samples.petclinic;

-import org.junit.Test;
-import org.junit.runner.RunWith;
+import org.junit.jupiter.api.Test;

 import org.springframework.beans.factory.annotation.Autowired;
 import org.springframework.boot.test.context.SpringBootTest;
 import org.springframework.samples.petclinic.vet.VetRepository;
-import org.springframework.test.context.junit4.SpringRunner;

-@RunWith(SpringRunner.class)
 @SpringBootTest
 public class PetclinicIntegrationTests {

@@ -32,7 +29,7 @@ public class PetclinicIntegrationTests {
     private VetRepository vets;

     @Test
-    public void testFindAll() throws Exception {
+    void testFindAll() throws Exception {
         vets.findAll();
         vets.findAll(); // served from cache
     }
```

Problem: Two objects are equal, although they should not be equal

```java
public final class Person {

    private String firstName;
    private String lastName;
    private String jobTitle;

    // more code like getter and setter

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Person person = (Person) o;
        return Objects.equals(firstName, person.firstName) &&
                Objects.equals(lastName, person.lastName) &&
                Objects.equals(jobTitle, person.jobTitle);
    }

    @Override
    public int hashCode() {
        return Objects.hash(firstName, lastName, jobTitle);
    }
}
```

equals() Contract

- It is *reflexive*: for any non-null reference value `x`, `x.equals(x)` should return `true`.
- It is *symmetric*: for any non-null reference values `x` and `y`, `x.equals(y)` should return `true` if and only if `y.equals(x)` returns `true`.
- It is *transitive*: for any non-null reference values `x`, `y`, and `z`, if `x.equals(y)` returns `true` and `y.equals(z)` returns `true`, then `x.equals(z)` should return `true`.
- It is *consistent*: for any non-null reference values `x` and `y`, multiple invocations of `x.equals(y)` consistently return `true` or consistently return `false`, provided no information used in `equals` comparisons on the objects is modified.
- For any non-null reference value `x`, `x.equals(null)` should return `false`.

As a reminder

hashCode() Contract

- hashCode() must return same integer consistently for the same object.
- hashCode() for two a.equals(b) object must be same, ie a.hashCode() is same as b.hashCode().
- hashCode() for two not equals() object may not be distinct.

As a reminder

EqualsVerifier

```
1  @Test
2  void equalsContractVerySimple(){
3      Person person1 = createPerson();
4      Person person2 = createPerson();
5      person2.setJobTitle("Consultant");
6
7      assertThat(person1).isNotEqualTo(person2);
8      assertThat(person1.hashCode()).isNotEqualTo(person2.hashCode());
9  }
```

```
12 @Test
13 void equalsContractMuchBetter(){
14     EqualsVerifier.forClass(Person.class)
15             .suppress(Warning.NONFINAL_FIELDS).verify();
16 }
```

```
java.lang.AssertionError: EqualsVerifier found a problem in class Person.
-> Mutability: equals depends on mutable field firstName.

For more information, go to: http://www.jqno.nl/equalsverifier/errormessages

    at nl.jqno.equalsverifier.EqualsVerifierApi.verify(EqualsVerifierApi.java:305)
    at com.github.sparsick.test.tool.database.PersonTest.equalsContractMuchBetter(PersonTest.java:26)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
    at
java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.base/java.lang.reflect.Method.invoke(Method.java:566)
    at org.junit.platform.commons.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:675)
    at org.junit.jupiter.engine.execution.MethodInvocation.proceed(MethodInvocation.java:60)
    at
org.junit.jupiter.engine.execution.InvocationInterceptorChain$ValidatingInvocation.proceed(InvocationIntercep
torChain.java:125)
    at org.junit.jupiter.engine.extension.TimeoutExtension.intercept(TimeoutExtension.java:132)
    at org.junit.jupiter.engine.extension.TimeoutExtension.interceptTestableMethod(TimeoutExtension.java:124)
    at org.junit.jupiter.engine.extension.TimeoutExtension.interceptTestMethod(TimeoutExtension.java:74)
    at
org.junit.jupiter.engine.execution.ExecutableInvoker$ReflectiveInterceptorCall.lambda$ofVoidMethod$0(Executab
leInvoker.java:115)
    at org.junit.jupiter.engine.execution.ExecutableInvoker.lambda$invoke$0(ExecutableInvoker.java:105)
    at
org.junit.jupiter.engine.execution.InvocationInterceptorChain$InterceptedInvocation.proceed(InvocationInterce
ptorChain.java:104)
```

What about *toString()?*

Problem: *toString()* does not include all class fields

```java
public final class Person {

    private String firstName;
    private String lastName;
    private String jobTitle;

    // more code like getter, setter, hashcode and equals


    @Override
    public String toString() {
        return "Person{" +
                "firstName='" + firstName + '\'' +
                ", lastName='" + lastName + '\'' +
                ", jobTitle='" + jobTitle + '\'' +
                '}';
    }
}
```

To String Verifier

```
1  @Test
2  void toStringVerifier(){
3      ToStringVerifier.forClass(Person.class)
4              .withClassName(NameStyle.SIMPLE_NAME)
5              .verify();
6  }
```

```
java.lang.AssertionError:

Failed verification:
com.github.sparsick.test.tool.database.Person

Expected auto generated toString:
Person{firstName='4b2b5d96-7d9f-46a3-a7ce-dd416c46706c', lastName='bb28f281-5cf4-43be-90ba-9dffaac274e6',
jobTitle='a000c70c-1107-4a30-b9b9-5176222a4da6'}

To start with class name: com.github.sparsick.test.tool.database.Person

    at com.jparams.verifier.tostring.ToStringVerifier.verify(ToStringVerifier.java:362)
    at com.github.sparsick.test.tool.database.PersonTest.toStringVerifier(PersonTest.java:33)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
    at
java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.base/java.lang.reflect.Method.invoke(Method.java:566)
    at org.junit.platform.commons.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:675)
    at org.junit.jupiter.engine.execution.MethodInvocation.proceed(MethodInvocation.java:60)
    at
org.junit.jupiter.engine.execution.InvocationInterceptorChain$ValidatingInvocation.proceed(InvocationIntercep
torChain.java:125)
    at org.junit.jupiter.engine.extension.TimeoutExtension.intercept(TimeoutExtension.java:132)
    at org.junit.jupiter.engine.extension.TimeoutExtension.interceptTestableMethod(TimeoutExtension.java:124)
    at org.junit.jupiter.engine.extension.TimeoutExtension.interceptTestMethod(TimeoutExtension.java:74)
    at
org.junit.jupiter.engine.execution.ExecutableInvoker$ReflectiveInterceptorCall.lambda$ofVoidMethod$0(Executab
```

To String Verifier

- NameStyle
- Preset
  - Eclipse
  - Intellji
  - Guava
  - ApacheToStringBuilder
- More Verifier (Extract)
  - withHashCode
  - withIgnoreFields
  - withNullField
  - withOnlyTheseFields

Problem: How should I test concurrency?

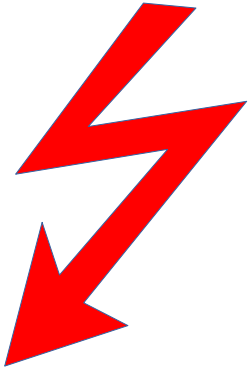
Awaitility

- Java DSL to synchronize asynchronous operations
- Support:
  - Groovy DSL
  - Kotlin DSL
  - Scala DSL

Import !

Awaitility does nothing to ensure thread safety or thread synchronization.

It‘s your job!



Sample

```
@Test
void awaitilityDemo() {
    var demoUnderTest = new ConcurrencyDemo();
    demoUnderTest.addItem("Hello World");

    await().until(demoUnderTest::hasNewItem);
    assertThat(demoUnderTest.allItems()).contains("Hello World");
}
```

More Possibilities (Excerpt)

Querying fields:

```
await().until( fieldIn(object).ofType(int.class), equalTo(2) );
```

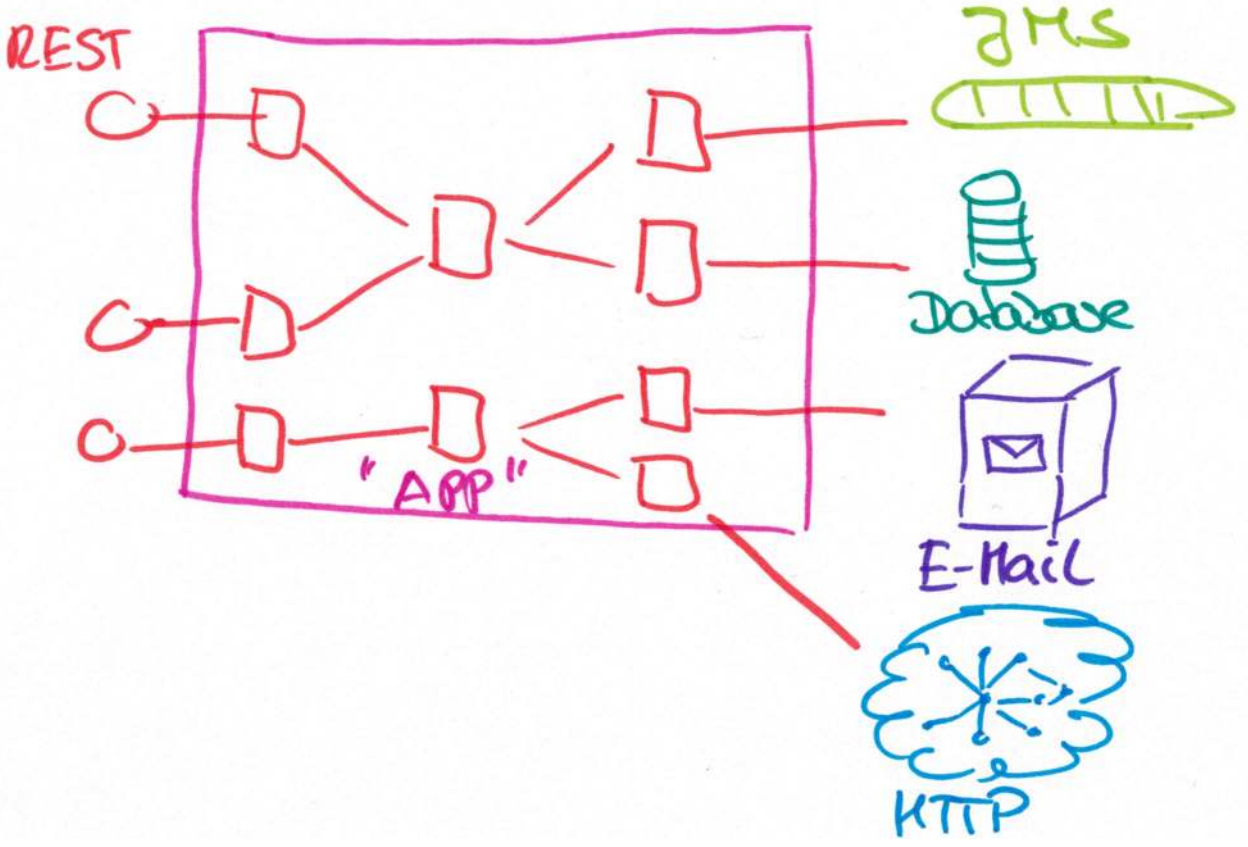
Ignore exceptions:

```
given().ignoreExceptions().await().until(() -> SocketUtils.findAvailableTcpPort(x,y));
```

Waiting for a minimum time:

```
await().atLeast(1, SECONDS).and().atMost(2, SECONDS).until(value(), equalTo(1));
```

Problem: Integrated Tests


REST
JMS
Database
"APP"
E-Mail
HTTP

What is the problem?

- Interaction with intrastructure are tested very late
  - Feedback in case of errors very late
- You are dependent on a special infrastructure
  - Sometimes during the build (Bad smell)
  - False negative error quote very high
  - Complex setup
  - Test execution slow

Integration vs Integrated Tests

Integrated Tests

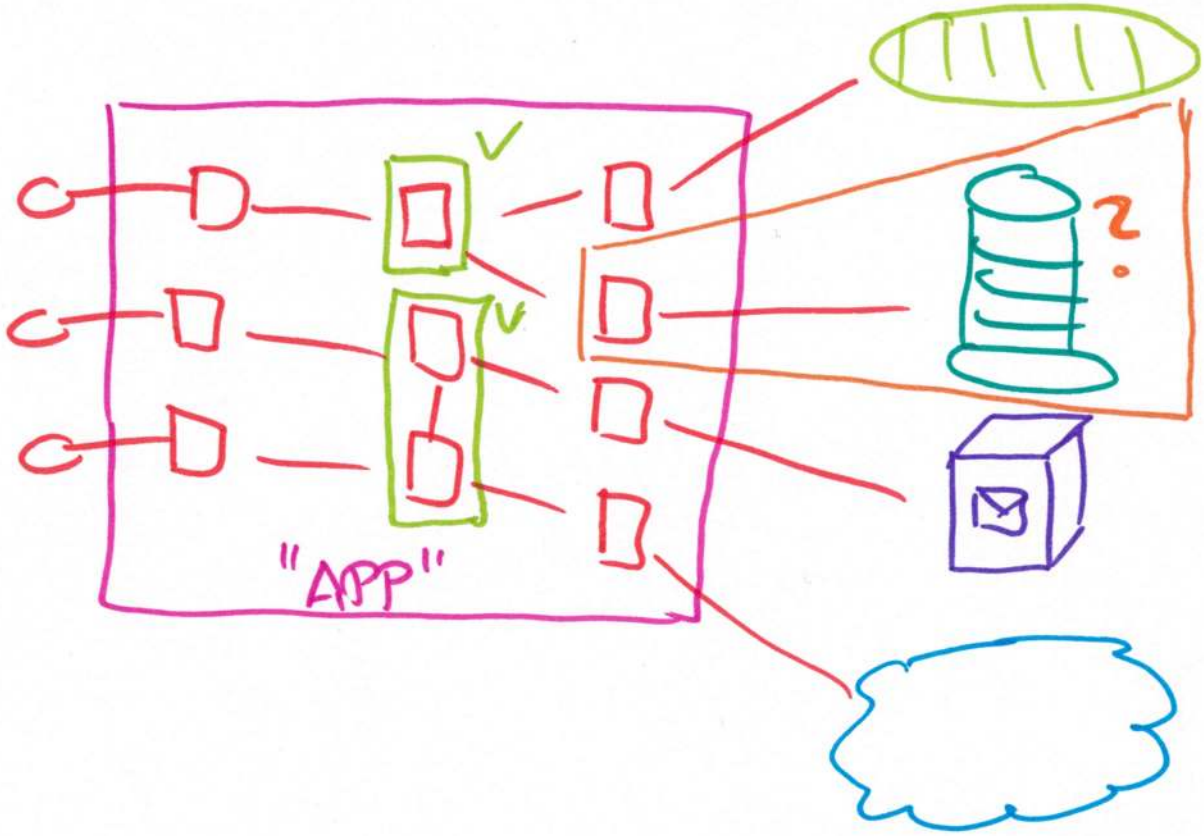
A test that will pass or fail based on the correctness of another system.

J.B.Rainsberger

Signs for having Integrated Tests:

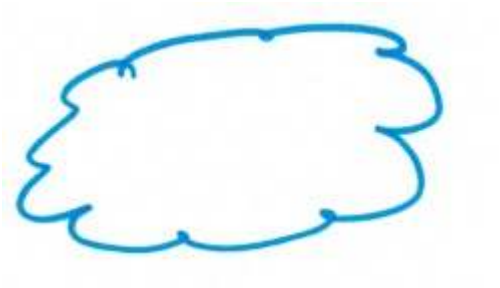
- We spin up other services in a local testing environment
- We test against other services in a shared testing environment
- Changes to your system breaks tests for other systems

from Spotify Blog Post


V
V
"APP"
?

Requirements

- Build run has to be independented on other systems
  - Minimize false negative errors
  - Decoupling between developers
- Feedback in case of error as soon as possible
- Speed test execution up

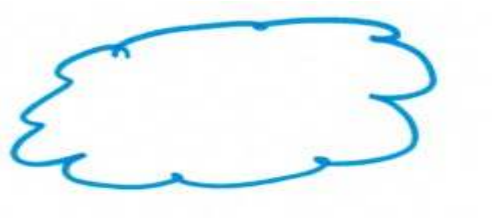


HTTP

E-Mail

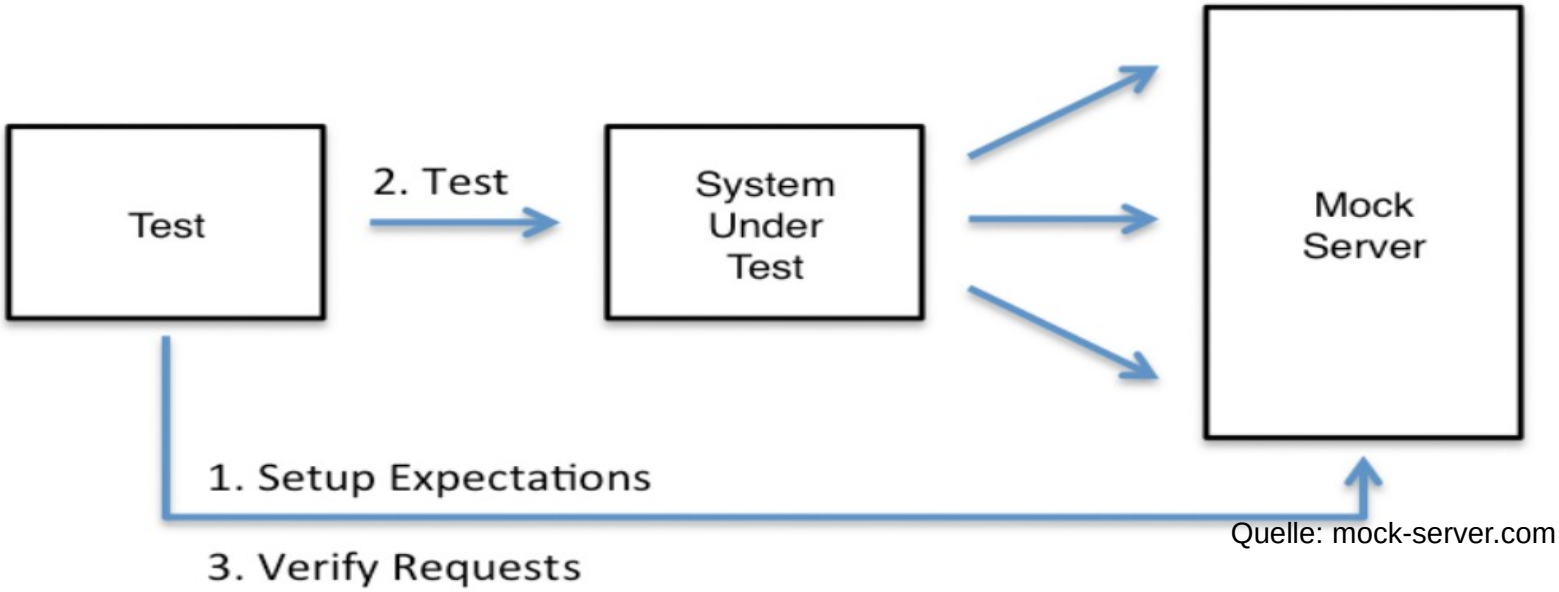
Datenbank

HTTP



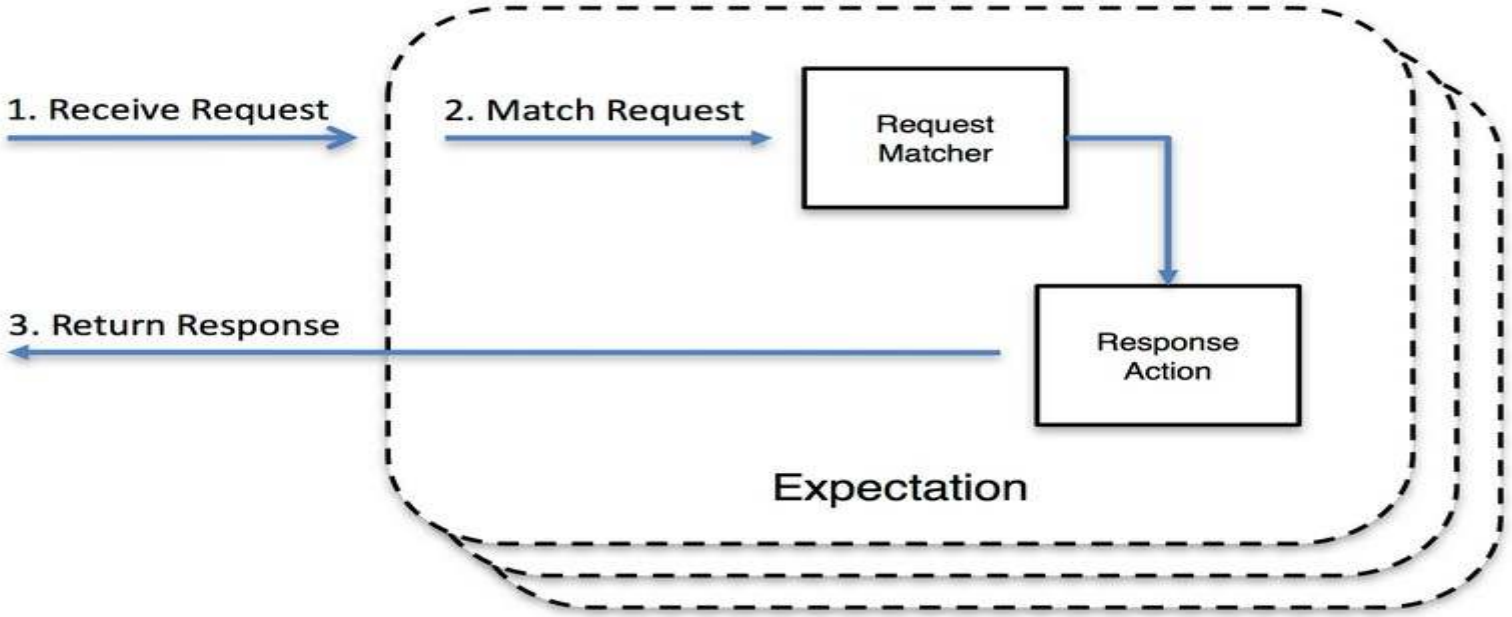
Mocking HTTP Calls

- MockServer (http://mock-server.com/)
- WireMock (http://wiremock.org/)



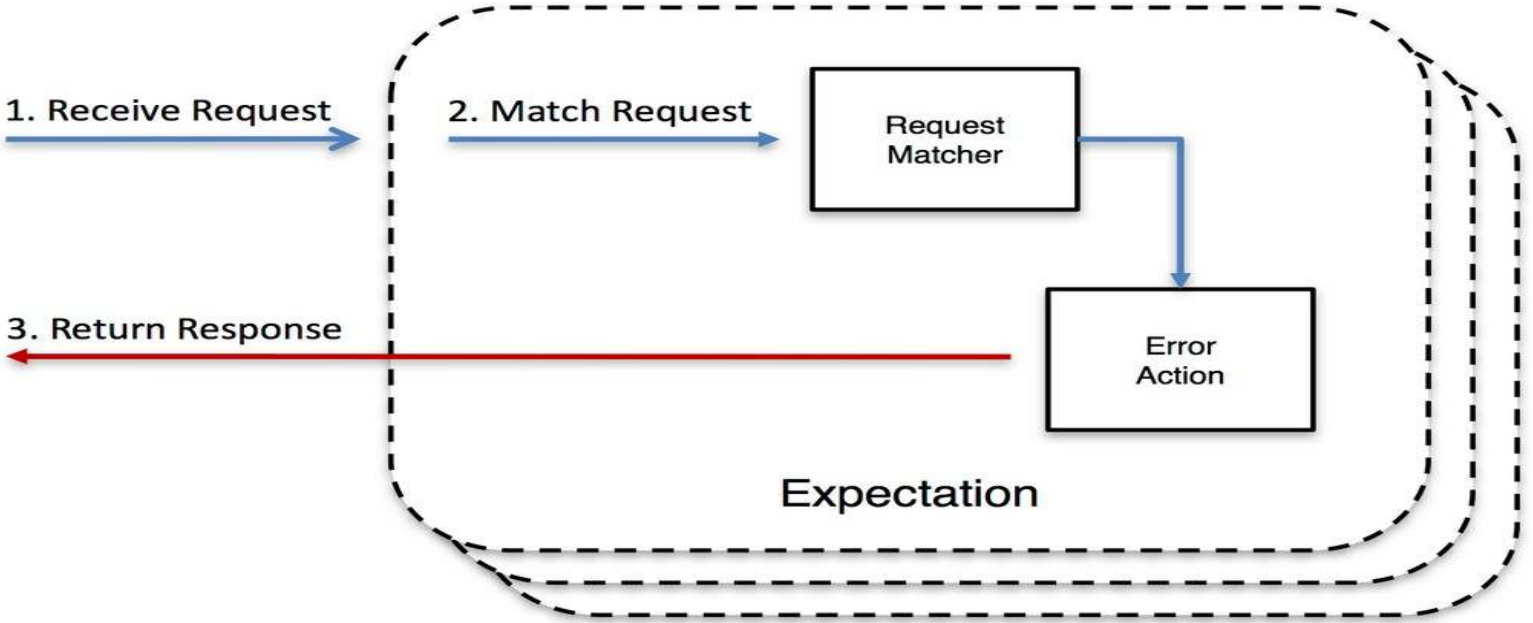
Quelle: mock-server.com

MockServer



Quelle: mock-server.com

MockServer



Quelle: mock-server.com

```
{
  "count": 37,
  "next": "https://swapi.co/api/starships/?page=2",
  "previous": null,
  "results": [
    {
      "name": "Executor",
      "model": "Executor-class star dreadnought",
      "manufacturer": "Kuat Drive Yards, Fondor Shipyards",
      "cost_in_credits": "1143350000",
      "length": "19000",
      "max_atmosphering_speed": "n/a",
      "crew": "279144",
      "passengers": "38000",
      "cargo_capacity": "250000000",
      "consumables": "6 years",
      "hyperdrive_rating": "2.0",
      "MGLT": "40",
      "starship_class": "Star dreadnought",
      "pilots": [],
      "films": [
        "https://swapi.co/api/films/2/",
        "https://swapi.co/api/films/3/"
      ],
      "created": "2014-12-15T12:31:42.547000Z",
      "edited": "2017-04-19T10:56:06.685592Z",
      "url": "https://swapi.co/api/starships/15/"
    },
```


http://swapi.co/api/starships

```java
public class StarWarsClient {
    private final RestTemplate restTemplate;
    private final String baseUrl;

    public StarWarsClient(String baseUrl) {
        this.restTemplate = new RestTemplateBuilder().build();
        try {
            this.baseUrl = new URL(baseUrl).toString();
        } catch (MalformedURLException e) {
            throw new RuntimeException(e);
        }
    }

    public List<Starship> findAllStarships() {
        List<Starship> starships = new ArrayList<>();
        String nextPageUrl = baseUrl + "/api/starships";
        do {
            String forObject = restTemplate.getForObject(nextPageUrl, String.class);
            Map<String, Object> jsonMap = new GsonJsonParser().parseMap(forObject);
            nextPageUrl = (String) jsonMap.get("next");
            for (Map result : (List<Map>) jsonMap.get("results")) {
                starships.add(Starship.from(result));
            }
        }
        while (nextPageUrl != null);
        return starships;
    }
}
```

```java
@ExtendWith(MockServerExtension.class)
public class StarWarsClientMockserverTest {

    private MockServerClient mockServerClient;
    private StarWarsClient clientUnderTest;

    public StarWarsClientMockserverTest(MockServerClient mockServerClient) {
        this.mockServerClient = mockServerClient;
        clientUnderTest = new StarWarsClient("http://localhost:" + mockServerClient.remoteAddress().getPort());
    }

    @Test
    void findAllStarships() {
        mockServerClient
                .when(request()
                        .withMethod("GET")
                        .withPath("/api/starships")
                )
                .respond(response()
                        .withBody(starshipTestData)
                );
        mockServerClient
                .when(request()
                        .withMethod("GET")
                        .withPath("/api/starships2")
                )
                .respond(response()
                        .withBody(starshipTestData2)
                );

        List<Starship> allStarships = clientUnderTest.findAllStarships();

        assertThat(allStarships).hasSize(11);
    }
}
```

Don’t mock API you don’t own

Verified Fakes

Tests, that run against real API und verify if test doubles are still correct

Own HTTP Endpoints

- Jersey → JerseyTest
- Spring MVC → MockMvc
- Framework-indepentent:
  - REST Assured (http://rest-assured.io/)
    - „Killer“-Feature: JsonPath, XmlPath
    - Scala Support
    - Spring MVC Support
    - Given-when-then structure

Rest Assured - JsonPath

```
{
"lotto":{
 "lottoId":5,
 "winning-numbers":[2,45,34,23,7,5,3],
 "winners":[{
   "winnerId":23,
   "numbers":[2,45,34,23,3,5]
 },{
   "winnerId":54,
   "numbers":[52,3,12,11,18,22]
 }]
}
}
```

http://localhost:8080/lotto

```
get("/lotto").then().body("lotto.lottoId", equalTo(5));
```

```
get("/lotto").then().body("lotto.winners.winnerId", hasItems(23, 54));
```

Quelle: Rest Assured Doc

Rest Assured - XMLPath

```
<greeting>
   <firstName>{params("firstName")}</firstName>
   <lastName>{params("lastName")}</lastName>
</greeting>
```

```
given().
        parameters("firstName", "John", "lastName", "Doe").
when().
        post("/greetXML").
then().
        body("greeting.firstName", equalTo("John")).
        body("greeting.lastName", equalTo("Doe"));
```

http://localhost:8080/greetXML

Quelle: Rest Assured Doc

Rest Assured – Spring MVC

```java
@Controller
@RequestMapping("starwars")
public class StarWarsMovieController {

    @RequestMapping(value="movies", method = RequestMethod.GET, produces = "application/json")
    public @ResponseBody List<StarWarsMovie> findAllMovies(){
        return List.of(new StarWarsMovie("A New Hope", "George Lucas"),
                new StarWarsMovie("The Force Awakens", "J. J. Abrams"));
    }
}
```

Rest Assured – Spring MVC

```
@RunWith(SpringRunner.class)
@WebMvcTest(StarWarsMovieController.class)
public class StarWarsMovieControllerITest {

    @Autowired
    private MockMvc mockMvc;

    @Test
    public void findAllMovies(){
        given().mockMvc(mockMvc).
        when().get("/starwars/movies").
        then().statusCode(200)
                .body("title",  hasItems("A New Hope", "The Force Awakens"))
                .body("director", hasItems("George Lucas", "J. J. Abrams"));
    }

    @Test
    public void findAllMovies_plainMockMvc() throws Exception {
        mockMvc.perform(get("/starwars/movies"))
                .andExpect(status().isOk())
                .andExpect(jsonPath("[0].title").value("A New Hope"))
                .andExpect(jsonPath("[0].director").value("George Lucas"))
                .andExpect(jsonPath("[1].title").value("The Force Awakens"))
                .andExpect(jsonPath("[1].director").value("J. J. Abrams"));
    }
}
```

Unit Tests for Endpoints

```
class StarWarsMovieControllerTest {

    @Test
    void findAllMovies(){
        StarWarsMovieController controllerUnderTest = new StarWarsMovieController();
        List<StarWarsMovie> movies = controllerUnderTest.findAllMovies();

        assertThat(movies)
                .hasSize(2)
                .extracting("director")
                .containsExactlyInAnyOrder( "George Lucas","J. J. Abrams" );
    }
}
```

E-Mail

E-Mail

- GreenMail (http://www.icegreen.com/greenmail/)
  - Lightweight sand boxed mail server
    - Sending and Retrieving
  - SMTP, POP3, IMAP incl. SSL
  - JUnit5 integration
  - Usage model:
    - Java API
    - Standalone
    - WAR
    - Docker Container

Greenmail

```
public class MailClient {

    private final JavaMailSender javaMailSender;

    public MailClient(JavaMailSender javaMailSender) {
        this.javaMailSender = javaMailSender;
    }

    public void sendMessage(String from, String to, String body) throws MessagingException {
        MimeMessage message = javaMailSender.createMimeMessage();
        message.setFrom(new InternetAddress(from));
        message.addRecipient(Message.RecipientType.TO, new InternetAddress(to));
        message.setText(body);
        javaMailSender.send(message);
    }
}
```

```java
class MailClientTest {

    private static int smtpPort = SocketUtils.findAvailableTcpPort();
    private static final ServerSetup SERVER_SETUP_SMTP =
                    new ServerSetup(smtpPort, null, ServerSetup.PROTOCOL_SMTP);

    @RegisterExtension
    static GreenMailExtension greenMail = new GreenMailExtension(new ServerSetup[]{
            SERVER_SETUP_SMTP});


    private MailClient clientUnderTest;

    @AfterEach
    void cleanUp(){ greenMail.reset(); }

    @BeforeEach
    void setUp() {
        Properties properties = createProperties();
        clientUnderTest = new MailClient(properties, "test", "xxx");
    }

    @Test
    void sendMessage() throws MessagingException, IOException {
        String message = "text";
        String from = "user@example.com";
        String to = "to@example.com";
        clientUnderTest.sendMessage(from, to, message);

        MimeMessage[] receivedMessages = greenMail.getReceivedMessages();
        assertThat(receivedMessages).hasSize(1);
        String content = (String) receivedMessages[0].getContent();
        assertThat(content.trim()).isEqualTo(message);
    }

    private Properties createProperties() {
}
```

Databases

Databases

- Embedded databases
  - H2, Derby
  - Not production-related
  - Not everything testable
- Standalone databases
  - Dependence on specific infrastructure
- Shared databases
  - Close coupling between developers
  - High false negative error rate


TESTCONTAINERS

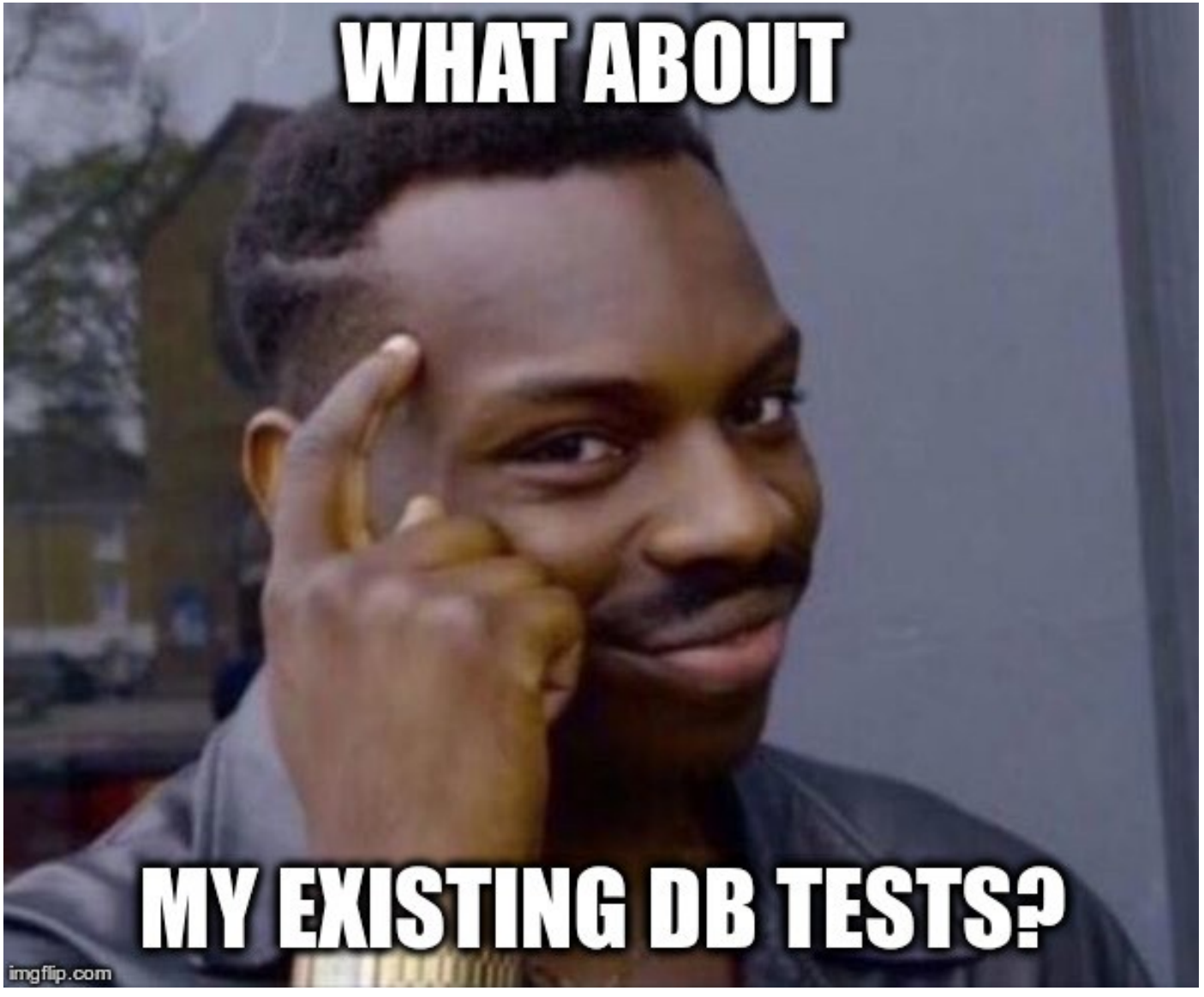
```java
@Testcontainers
class PersonRepositoryJUnit5Test {

    @Container
    private PostgreSQLContainer postgres = new PostgreSQLContainer();

    private PersonRepository repositoryUnderTest;

    @BeforeEach
    void setup(){
        HikariConfig hikariConfig = new HikariConfig();
        hikariConfig.setJdbcUrl(postgres.getJdbcUrl());
        hikariConfig.setUsername(postgres.getUsername());
        hikariConfig.setPassword(postgres.getPassword());

        HikariDataSource ds = new HikariDataSource(hikariConfig);
        Flyway flyway = Flyway.configure().dataSource(ds).load();
        flyway.migrate();

        repositoryUnderTest = new PersonRepository(ds);
    }

    @Test
    void saveAndFindAllPerson() {
       Person person = new Person();
       person.setFirstName("firstName");
       person.setLastName("lastName");
       person.setJobTitle("jobTitle");

       repositoryUnderTest.save(person);

        List<Person> persons = repositoryUnderTest.findAllPersons();
        assertThat(persons).hasSize(1).contains(person);
    }
}
```

```
@Testcontainers
class DbMigrationJUnit5Test {

    @Container
    private MySQLContainer mysqlDb = new MySQLContainer();

    @Test
    void testDbMigrationFromTheScratch(){
        Flyway flyway = Flyway.configure()
                        .dataSource(mysqlDb.getJdbcUrl(),
                                    mysqlDb.getUsername(),
                                    mysqlDb.getPassword()).load();

        flyway.migrate();
    }
}
```

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running db.migration.DbMigrationITest
INFO  - ertyClientProviderStrategy - Found docker client settings from environment
INFO  - ckerClientProviderStrategy - Found Docker environment with Environment variables, system properties and defaults. Resolved:
    dockerHost=unix:///var/run/docker.sock
    apiVersion='{UNKNOWN_VERSION}'
    registryUrl='https://index.docker.io/v1/'
    registryUsername='sparsick'
    registryPassword='null'
    registryEmail='null'
    dockerConfig='DefaultDockerClientConfig[dockerHost=unix:///var/run/docker.sock,registryUsername=sparsick,registryPassword=<null>,registryEma

INFO  - DockerClientFactory        - Docker host IP address is localhost
INFO  - DockerClientFactory        - Connected to docker:
  Server Version: 17.05.0-ce
  API Version: 1.29
  Operating System: Linux Mint 18.2
  Total Memory: 19511 MB
        ℹ Checking the system...
        ✔ Docker version is newer than 1.6.0
        ✔ Docker environment has more than 2GB free
        ✔ File should be mountable
        ✔ Exposed port is accessible
INFO  - 🐳 [mysql:latest]          - Creating container for image: mysql:latest
INFO  - 🐳 [mysql:latest]          - Starting container with ID: 2668be66c2631e49b5bcb4e180665d223525ec896ea78034326076d5f9063d53
INFO  - 🐳 [mysql:latest]          - Container mysql:latest is starting: 2668be66c2631e49b5bcb4e180665d223525ec896ea78034326076d5f9063d53
INFO  - 🐳 [mysql:latest]          - Waiting for database connection to become available at jdbc:mysql://localhost:32769/test using query 'SELECT
INFO  - 🐳 [mysql:latest]          - Obtained a connection to container (jdbc:mysql://localhost:32769/test)
INFO  - 🐳 [mysql:latest]          - Container mysql:latest started
INFO  - VersionPrinter             - Flyway 4.0.3 by Boxfuse
INFO  - DbSupportFactory           - Database: jdbc:mysql://localhost:32769/test (MySQL 5.7)
INFO  - DbValidate                 - Successfully validated 2 migrations (execution time 00:00.011s)
INFO  - MetaDataTableImpl          - Creating Metadata table: `test`.`schema_version`
INFO  - DbMigrate                  - Current version of schema `test`: << Empty Schema >>
INFO  - DbMigrate                  - Migrating schema `test` to version 1.0.0 - create person table
INFO  - DbMigrate                  - Migrating schema `test` to version 2.0.0 - add column job title
INFO  - DbMigrate                  - Successfully applied 2 migrations to schema `test` (execution time 00:00.133s).
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 13.9 sec
```

Testcontainers

- Temporary database containers - special MySQL, PostgreSQL, Oracle XE and Virtuoso container
- Webdriver containers - Dockerized Chrome or Firefox browser for Selenium/Webdriver operation with automated video recording
- Further special container – Elasticsearch, Kafka, Apache Pulsar, Mockserver, Toxiproxy, Nginx, Hashicorp Vault
- Generic containers – any Docker container
- Docker compose – Reuse of Docker Compose YAML Datei
- Dockerfile containers – Container directly from Dockerfile


WHAT ABOUT
MY EXISTING DB TESTS?
imgflip.com

Container Via JDBC URL

```
public class PersonRepositoryJdbcUrlTestContainerTest {
    private PersonRepository repositoryUnderTest;
    private Flyway flyway;

    @BeforeEach
    void setup(){
        HikariConfig hikariConfig = new HikariConfig();
        hikariConfig.setJdbcUrl("jdbc:tc:postgresql:9.6.8:///persondb");
        hikariConfig.setUsername("postgres");
        hikariConfig.setPassword("");

        DataSource ds = new HikariDataSource(hikariConfig);
        flyway = Flyway.configure().dataSource(ds).load();
        flyway.migrate();

        repositoryUnderTest = new PersonRepository(ds);

    }

    @AfterEach
    void cleanUp(){
        flyway.clean();
    }
}
```

Docker Maven Plugin (DMP)

- fabric8io/docker-maven-plugin (http://dmp.fabric8.io)
- Well known for building Docker Image
- It also can start and stop container

Maven Phase: Integration-Test

- Starting Point:
  - Integrated tests against embedded or standalone database exist
- Improvement:
  - DMP starts container before test execution (*pre-integration-test*)
  - DMP stops container after test execution (*post-integration-test*)

Summary

Integrated Tests
Mockserver, Wiremock,
REST Assured
Greenmail
Testcontainers

equals, hashCode, toString
EqualsVerifier, ToStringVerifier

Well readable Assertions
AssertJ, JUnit5 Group Assertion

Concurrency
Awaitility

Test Data
generatedata.com
JavaFaker
ObjectMother, TestDataBuilder

Parametrized
Tests
JUnit5, Spock

Tests that run only under certain
conditions
JUnit5 ConditionsTest

Questions?

mail@sandra-parsick.de
@SandraParsick
@sparsick@mastodon.social

https://github.com/sparsick/test-tool-talk

More Information

- https://www.martinfowler.com/bliki/ObjectMother.html
- http://natpryce.com/articles/000714.html
- http://coding-is-like-cooking.info/2018/04/pre-tested-integration-back-to-the-basis-of-ci/
- http://blog.thecodewhisperer.com/permalink/integrated-tests-are-a-scam
- http://blog.thecodewhisperer.com/permalink/clearing-up-the-integrated-tests-scam
- https://bee42.com/de/blog/The_dark_age_of_container_testing/

More Information

- https://labs.spotify.com/2018/01/11/testing-of-microservices/
- https://martinfowler.com/bliki/IntegrationTest.html
- https://codewithoutrules.com/2016/07/31/verified-fakes/