Kapitola 1. Theories

Obsah

- je to rozšíření možností parametrizovatelných testů o možnost předat parametry testu jako skutečné parametry
- data, s nimiž má být test spouštěn, jsou označena anotací `@DataPoint`
  - rozlišuje se jejich deklarovaný typ – musí být objektový, nikoliv např. `int`
- metoda, která je označená anotací `@Theory`, je `public void` metoda s jedním nebo více parametry
- při spuštění testovací třídy označené jako `@RunWith(Theories.class)` se opakovaně zavolá metoda s anotací `@Theory`
  - její skutečné parametry budou data v jednotlivých `@DataPoint`, která typově vyhovují formálním parametrům metody

```
package theories;

import org.junit.experimental.theories.DataPoint;
import org.junit.experimental.theories.Theories;
import org.junit.experimental.theories.Theory;
import org.junit.runner.RunWith;

@RunWith(Theories.class)
public class TheoriesTests_1 {

  @DataPoint
  public static String DOBRE_JIDLO_1 = "pivo, knedlo, vepřo, zelo";
  @DataPoint
  public static String DOBRE_JIDLO_2 = "knedlo, vepřo, zelo, pivo";
  @DataPoint
  public static String JIDLO = "cola, hranolky";
  @DataPoint
  public static Integer DOBRA_CENA = new Integer(99);
  @DataPoint
  public static Integer CENA = new Integer(300);

  @Theory
  public void spravneSlozeniAPoradiJidla(String jidelniListek) {
    System.err.println("Testuje se: " + jidelniListek);
  }

  @Theory
  public void vyhovujiciCena(Integer cenaJidla) {
    System.err.println("Testuje se: " + cenaJidla);
```

```
  }
}
```

- spuštěn známým runnerem vypíše:

```
Testuje se: 99
Testuje se: 300
Testuje se: pivo, knedlo, vepřo, zelo
Testuje se: knedlo, vepřo, zelo, pivo
Testuje se: cola, hranolky
Výsledek všech testů: true
Doba běhu všech testů [ms]: 50
Počet spuštěných testů: 2
Počet ignorovaných testů: 0
Počet testů, které selhaly: 0
```

- pokud do metod anotovaných `@Theory` přidáme nějaký test, bude se tato metoda provádět tak dlouho, dokud na některých datech z `@DataPoint` neselže
  - v tomto případě selhává celý test a v pořadí následující `@DataPoint` již nejsou do metody testu předávány

```
package theories;

import static org.hamcrest.CoreMatchers.startsWith;
import static org.junit.Assert.assertThat;
import static org.junit.Assert.assertTrue;

import org.junit.experimental.theories.DataPoint;
import org.junit.experimental.theories.Theories;
import org.junit.experimental.theories.Theory;
import org.junit.runner.RunWith;

@RunWith(Theories.class)
public class TheoriesTests_2 {

  @DataPoint
  public static String DOBRE_JIDLO_1 = "pivo, knedlo, vepřo, zelo";
  @DataPoint
  public static String JIDLO = "cola, hranolky";
  @DataPoint
  public static String DOBRE_JIDLO_2 = "knedlo, vepřo, zelo, pivo";

  @DataPoint
  public static Integer CENA = new Integer(300);
  @DataPoint
  public static Integer DOBRA_CENA = new Integer(99);

  @Theory
  public void spravneSlozeniAPoradiJidla(String jidelniListek) {
    System.err.println("Testuje se: " + jidelniListek);
    assertThat("Nevyhovující pořadí: ", jidelniListek, startsWith("pivo"));
  }
```

```
  @Theory
  public void vyhovujiciCena(Integer cenaJidla) {
    System.err.println("Testuje se: " + cenaJidla);
    assertTrue("Vysoká cena: ", (cenaJidla.intValue() < 100));
  }
}
```

- spuštěn známým runnerem vypíše:

```
Testuje se: 300
Testuje se: pivo, knedlo, vepřo, zelo
Testuje se: cola, hranolky
Výsledek všech testů: false
Doba běhu všech testů [ms]: 80
Počet spuštěných testů: 2
Počet ignorovaných testů: 0
Počet testů, které selhaly: 2
Výpis selhaných testů:
getTestHeader: vyhovujiciCena(theories.TheoriesTests_2)
getMessage: vyhovujiciCena(CENA)
getTestHeader: spravneSlozeniAPoradiJidla(theories.TheoriesTests_2)
getMessage: spravneSlozeniAPoradiJidla(JIDLO)
```

  - při hodnotě `300` selhal test v metodě `vyhovujiciCena()` a proto tato metoda není dále volána, ani pro následnou vyhovující cenu `99`
  - v případě metody `spravneSlozeniAPoradiJidla()` prošel test s prvním `@DataPoint` v pořadí (pivo, knedlo, vepřo, zelo), ale druhý neprošel (cola, hranolky) a proto na třetí v pořadí nedošlo

1.1. Assume v @Theory

- pokud je v metodě anotované `@Theory` před `assertXY()` podmínka `assumeXY()`, pak je nevyhovující `@DataPoint` ignorován a test v tom případě neselhává
- jestliže vyhoví `assumeXY()` pro všechny `@DataPoint` a některý následný `assertXY()` selže, pak selhává test

```
package theories;

import static org.hamcrest.CoreMatchers.containsString;
import static org.hamcrest.CoreMatchers.startsWith;
import static org.junit.Assert.assertThat;
import static org.junit.Assert.assertTrue;
import static org.junit.Assume.assumeThat;
import static org.junit.Assume.assumeTrue;

import org.junit.experimental.theories.DataPoint;
import org.junit.experimental.theories.Theories;
import org.junit.experimental.theories.Theory;
import org.junit.runner.RunWith;

@RunWith(Theories.class)
```

```
public class TheoriesTests_3 {

  @DataPoint
  public static String DOBRE_JIDLO_1 = "pivo, knedlo, vepřo, zelo";
  @DataPoint
  public static String JIDLO = "cola, hranolky";
  @DataPoint
  public static String DOBRE_JIDLO_2 = "knedlo, vepřo, zelo, pivo";

  @DataPoint
  public static Integer DOBRA_CENA = new Integer(99);
  @DataPoint
  public static Integer CENA = new Integer(300);

  @Theory
  public void spravneSlozeniAPoradiJidla(String jidelniListek) {
    System.err.println("Testuje se: " + jidelniListek);
   assumeThat("Nevyhovující složení: ", jidelniListek, containsString("pivo"));
    assertThat("Nevyhovující pořadí: ", jidelniListek, startsWith("pivo"));
  }

  @Theory
  public void vyhovujiciCena(Integer cenaJidla) {
    System.err.println("Testuje se: " + cenaJidla);
    assumeTrue("Neakceptovatelná cena: ", (cenaJidla.intValue() < 500));
    assertTrue("Vysoká cena: ", (cenaJidla.intValue() < 100));
  }
}
```

- spuštěn známým runnerem vypíše:

```
Testuje se: 99
Testuje se: 300
Testuje se: pivo, knedlo, vepřo, zelo
Testuje se: cola, hranolky
Testuje se: knedlo, vepřo, zelo, pivo
Výsledek všech testů: false
Doba běhu všech testů [ms]: 70
Počet spuštěných testů: 2
Počet ignorovaných testů: 0
Počet testů, které selhaly: 2
Výpis selhaných testů:
getTestHeader: vyhovujiciCena(theories.TheoriesTests_3)
getMessage: vyhovujiciCena(CENA)
getTestHeader: spravneSlozeniAPoradiJidla(theories.TheoriesTests_3)
getMessage: spravneSlozeniAPoradiJidla(DOBRE_JIDLO_2)
```

  - u testu ceny obě dvě ceny vyhověly `assumeTrue()`, ale test selhal při (druhé) ceně `300`
  - u testu složení a pořadí jídla nevyhověl `assumeThat()` v pořadí druhý (cola, hranolky) a byl z testu vynechán
    - proto mohl být testován v pořadí třetí (knedlo, vepřo, zelo, pivo), který však nevhodným pořadím shodil test

1.2. Více formálních parametrů v `@Theory`

- dosud měla testovací metoda anotovaná pomocí `@Theory` jen jeden formální parametr
- pokud jich bude mít více, vyvolá se tato metoda se všemi vyhovujícími kombinacemi z `@DataPoint`
  - to významně rozšiřuje škálu testů

```
package theories;

import static org.hamcrest.CoreMatchers.containsString;
import static org.hamcrest.CoreMatchers.startsWith;
import static org.junit.Assert.assertThat;
import static org.junit.Assert.assertTrue;
import static org.junit.Assume.assumeThat;
import static org.junit.Assume.assumeTrue;

import org.junit.experimental.theories.DataPoint;
import org.junit.experimental.theories.Theories;
import org.junit.experimental.theories.Theory;
import org.junit.runner.RunWith;

@RunWith(Theories.class)
public class TheoriesTests_4 {

  @DataPoint
  public static String DOBRE_JIDLO_1 = "pivo, knedlo, vepřo, zelo";
  @DataPoint
  public static String JIDLO = "cola, hranolky";
  @DataPoint
  public static String DOBRE_JIDLO_2 = "knedlo, vepřo, zelo, pivo";

  @DataPoint
  public static Integer DOBRA_CENA = new Integer(99);
  @DataPoint
  public static Integer SUPER_CENA = new Integer(50);

  @Theory
  public void spravneSlozeniAPoradiJidlaAVyhovujiciCena(String jidelniListek, ▶
Integer cenaJidla) {
    System.err.println("Testuje se: " + jidelniListek + " za: " + cenaJidla);
    assumeThat("Nevyhovující složení: ", jidelniListek, containsString("pivo"));
    assumeTrue("Neakceptovatelná cena: ", (cenaJidla.intValue() < 500));
    assertThat("Nevyhovující pořadí: ", jidelniListek, startsWith("pivo"));
    assertTrue("Vysoká cena: ", (cenaJidla.intValue() < 100));
  }
}
```

- spuštěn známým runnerem vypíše:

```
Testuje se: pivo, knedlo, vepřo, zelo za: 99
Testuje se: pivo, knedlo, vepřo, zelo za: 50
Testuje se: cola, hranolky za: 99
```

```
Testuje se: cola, hranolky za: 50
Testuje se: knedlo, vepřo, zelo, pivo za: 99
Výsledek všech testů: false
Doba běhu všech testů [ms]: 70
Počet spuštěných testů: 1
Počet ignorovaných testů: 0
Počet testů, které selhaly: 1
Výpis selhaných testů:
getTestHeader: ►
spravneSlozeniAPoradiJidlaAVyhovujiciCena(theories.TheoriesTests_4)
getMessage: spravneSlozeniAPoradiJidlaAVyhovujiciCena(DOBRE_JIDLO_2, ►
DOBRA_CENA)
```

1.3. Více formálních parametrů stejného typu v `@Theory`

- velmi zajímavou možností je metoda `@Theory` s dvěma a více formálními parametry stejného typu
- pak jsou automaticky provedeny všechny kombinace všech `@DataPoint`

```
package theories;

import static org.junit.Assert.assertTrue;
import static org.junit.Assume.assumeTrue;

import org.junit.experimental.theories.DataPoint;
import org.junit.experimental.theories.Theories;
import org.junit.experimental.theories.Theory;
import org.junit.runner.RunWith;

@RunWith(Theories.class)
public class TheoriesTests_5 {

  private static final int AKCEPTOVANY_ROZDIL = 20;

  @DataPoint
  public static Integer HODNOTA_1 = new Integer(5);
  @DataPoint
  public static Integer HODNOTA_2 = new Integer(15);
  @DataPoint
  public static Integer HODNOTA_3 = new Integer(25);
  @DataPoint
  public static Integer HODNOTA_4 = new Integer(35);
  @DataPoint
  public static Integer HODNOTA_5 = new Integer(45);

  @Theory
  public void odecitani(Integer prvni, Integer druhy) {
    System.err.println("Testuje se: " + prvni + " a: " + druhy);
    assumeTrue("Nevyhovující hodnota prvního: ", prvni.compareTo(druhy) > 0);
```

```
    assertTrue("Rozdíl není vyšší než: " + AKCEPTOVANY_ROZDIL,
                (prvni - druhy) < AKCEPTOVANY_ROZDIL);
  }
}
```

- spuštěn známým runnerem vypíše:

```
spuštěn známým runnerem vypíše:
Testuje se: 5 a: 5
Testuje se: 5 a: 15
Testuje se: 5 a: 25
Testuje se: 5 a: 35
Testuje se: 5 a: 45
Testuje se: 15 a: 5
Testuje se: 15 a: 15
Testuje se: 15 a: 25
Testuje se: 15 a: 35
Testuje se: 15 a: 45
Testuje se: 25 a: 5
Výsledek všech testů: false
Doba běhu všech testů [ms]: 70
Počet spuštěných testů: 1
Počet ignorovaných testů: 0
Počet testů, které selhaly: 1
Výpis selhaných testů:
getTestHeader: odecitani(theories.TheoriesTests_5)
getMessage: odecitani(HODNOTA_3, HODNOTA_1)
```

- zde byly testy, kde první má hodnotu `5` zcela přeskočeny, protože nevyhovují `assumeTrue()` – hodnota 5 je menší než jakákoliv další hodnota
- testy, kdy první je 15 a druhý 5 nebo 15 proběhnou, pro zbylé hodnoty druhého (25, 35, 45) opět nevyhoví `assumeTrue()`
- poslední test (25 a 5) ukončí celé provádění, protože nevyhoví `assertTrue()`