czechitas

Java 1 • lekce 7

Filip Jirsák

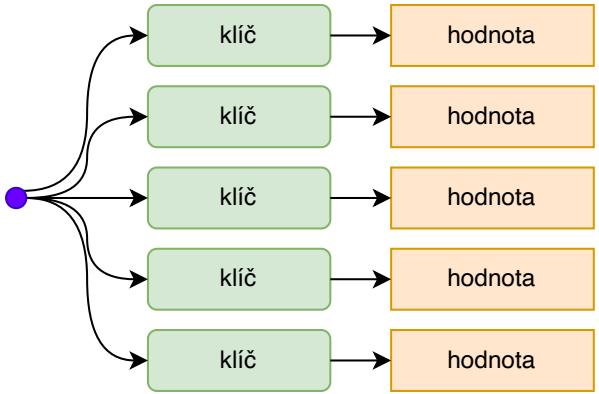
27. 2. 2024

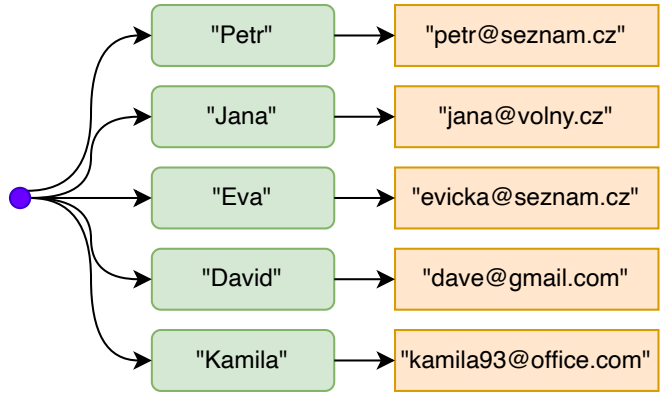
online

Mapa (slovník)

- seznam dvojic *klíč: hodnota*
- lze rychle hledat podle klíče
- nejpoužívanější `HashMap`

```
Map<Key, Value> map = new HahsMap<>();
```






```
Map<String, String> kontakty = new HahsMap<>();
kontakty.put("Jana", "jana@volny.cz");
String email = kontakty.get("Kamila");
// Proměnná email obsahuje "kamila93@office.com"
```

Automatizované testování (1)

- Náhrada za ruční opakované testování stále téhož
- Více testů
  - počítači je jedno, že testuje jednu věc 10× dokola s různými hodnotami
- Automatické spouštění při změnách
  - odhalení nezamýšlených dopadů změn
- Ověření případů, které „nemohou nastat“
- Lze měřit, jak velká část kódu je pokrytá testy

Automatizované testování (2)

- Vývojářské testy
  - součást zdrojového kódu
  - nejčastěji knihovna JUnit 5
- Testovací oddělení, Q&A
  - obvykle end-to-end testy
  - nástroje, které používají aplikaci „jako uživatel“
  - např. simulují klikání uživatele v aplikaci, zápis textu apod.

Knihovna JUnit (1)

- adresář `src/test/java`
- převážně jednotkové testy
  - testují jednu jednotku programu, tu nejmenší část, která dává samostatně smysl
- testovat náš kód, ne cizí
- testovat různé okrajové a speciální případy
- testovat chybové stavy

Knihovna JUnit (2)

- anotace `@Test`
- metody ve třídě `org.junit.jupiter.api.Assertions`
  - „očekávám, že výsledek bude…“
  - pořadí je vždy 1. očekávaná hodnota 2. skutečná hodnota
  - mnoho variant pro různé typy
- `assertEquals`
- `assertNotEquals`
- `assertNull`
- `assertNotNull`
- `assertTrue`
- `assertFalse`
- … a další

```
import static org.junit.jupiter.api.Assertions.*;

class TridaTest {

  @Test
  void testMetodyA() {
    …
  }

  @Test
  void testMetodyB() {
    …
  }

  @Test
  void testMetodyC() {
    …
  }

}
```

```java
@Test
void absKladne() {
  //volání testovaného kódu
  int vysledek = Math.abs(9);

  //ověření výsledků
  assertEquals(9, vysledek);
}

@Test
void absZaporne() {
  //volání testovaného kódu
  int vysledek = Math.abs(-11);

  //ověření výsledků
  assertEquals(11, vysledek);
}

@Test
void absNula() {
  //volání testovaného kódu
  int vysledek = Math.abs(0);

  //ověření výsledků
  assertEquals(0, vysledek);
}
```

```java
@Test
void testListu() {
  //volání testovaného kódu
  List<String> jmena = List.of("Lucie", "Dana", "Petra");

  //ověření výsledků
  assertEquals(3, jmena.size());
  assertEquals("Lucie", jmena.get(0));
  assertEquals("Dana", jmena.get(1));
  assertEquals("Petra", jmena.get(2));
  assertNull(jmena.get(3));
}
```