Testování aplikací

Ing. Jiří Mlejnek

Katedra softwarového inženýrství
Fakulta informačních technologií
České vysoké učení technické v Praze

jiri.mlejnek@fit.cvut.cz
Softwarové inženýrství BI-SI1
ZS 2015/ Před. 9

Evropský sociální fond
Praha & EU: Investujeme do vaší budoucnosti

Obsah

- Úvod
- Typy testů
- Jednotkové testy
- Testy komponent
- Integrační testy
- Systémové testy
- Statická analýza kódu

Úvod

- Cíle
  - Zajištění kvality
  - Měření kvality
  - Úspora financí / zdrojů

Úvod

- Rozsah testů
  - Kolik testů mám vytvářet?
  - Jak často mám testy provádět?
  - Kolik času věnovat testům?

Úvod

- Testovací případ
  - Postup provedení testu
  - Vstupní data
  - Výstupní data
  - Výsledek

Úvod

- Výsledek všech testů
  - Může pouze prokázat existenci chyb
  - Nemůže prokázat, že software neobsahuje chyby
- Pro zájemce o testování
  - Předmět BI-ATS

Úvod

Dotazy?

Typy testů

- Podle rozsahu
  - Jednotkové testy
  - Testy komponent
  - Integrační testy
  - Systémové testy

Typy testů

- Podle způsobu provádění
  - Statické
    - Nevyžadují spuštění programu
  - Dynamické
    - Vyžadují spustitelný kód

Typy testů

- Podle znalosti vnitřní struktury
  - White Box
    - Znalost zdrojového kódu
    - Umožňuje lépe identifikovat možné problémy
  - Black Box
    - Program je černá skříňka
    - Ověřuje se pouze její chování na rozhraní
  - Gray Box
    - Známé použité algoritmy, neznámá implementace

Typy testů

- Podle způsobu vyhodnocování
  - Automatické
  - Manuální

Typy testů

- Podle zaměření (kategorií) – FURPS
  - Funkčnost (Functionality)
  - Použitelnost (Usability) - BI-TUR
  - Spolehlivost (Reliability)
  - Výkon (Performance)
  - Podporovatelnost (Supportability)

Typy testů

Dotazy?

Jednotkové testy

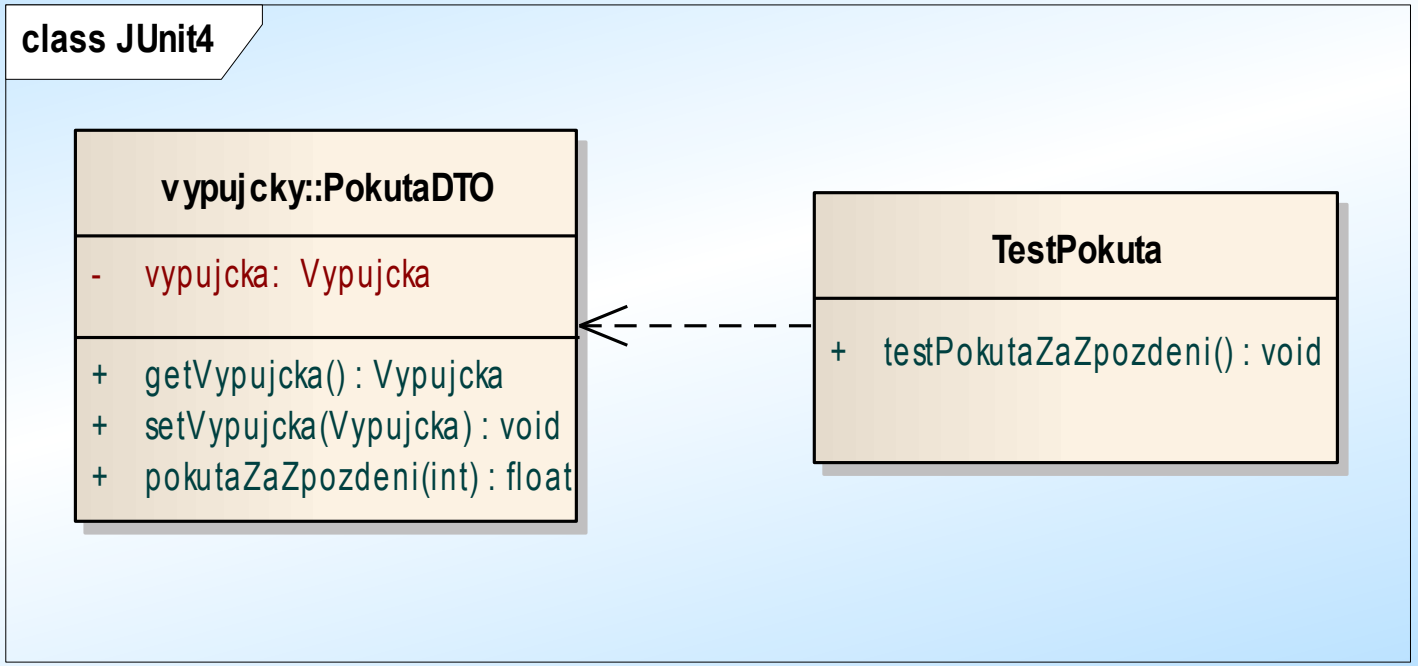
- Testování jednotlivých tříd
- Třídy jsou testovány samostatně
- Běží pouze krátkou dobu
- Nepoužívají žádný kontejner
- Lokálně spouštěny na počítači vývojáře

Jednotkové testy

- Nástroje
  - JUnit: http://www.junit.org/
  - NUnit: http://www.nunit.org/
  - Test NG: http://testng.org/

Jednotkové testy

- JUnit



Jednotkové testy

- JUnit 4

```
public class TestPokuta {
   @Test
   public void testPokutaZaZpozdeni(){
      PokutaDTO pokuta = new PokutaDTO();
      float vysePokuty = pokuta.pokutaZaZpozdeni(9);
      assertEquals(90,vysePokuty);
      vysePokuty = pokuta.pokutaZaZpozdeni(10);
      assertEquals(200,vysePokuty);
   }
}
```

Jednotkové testy

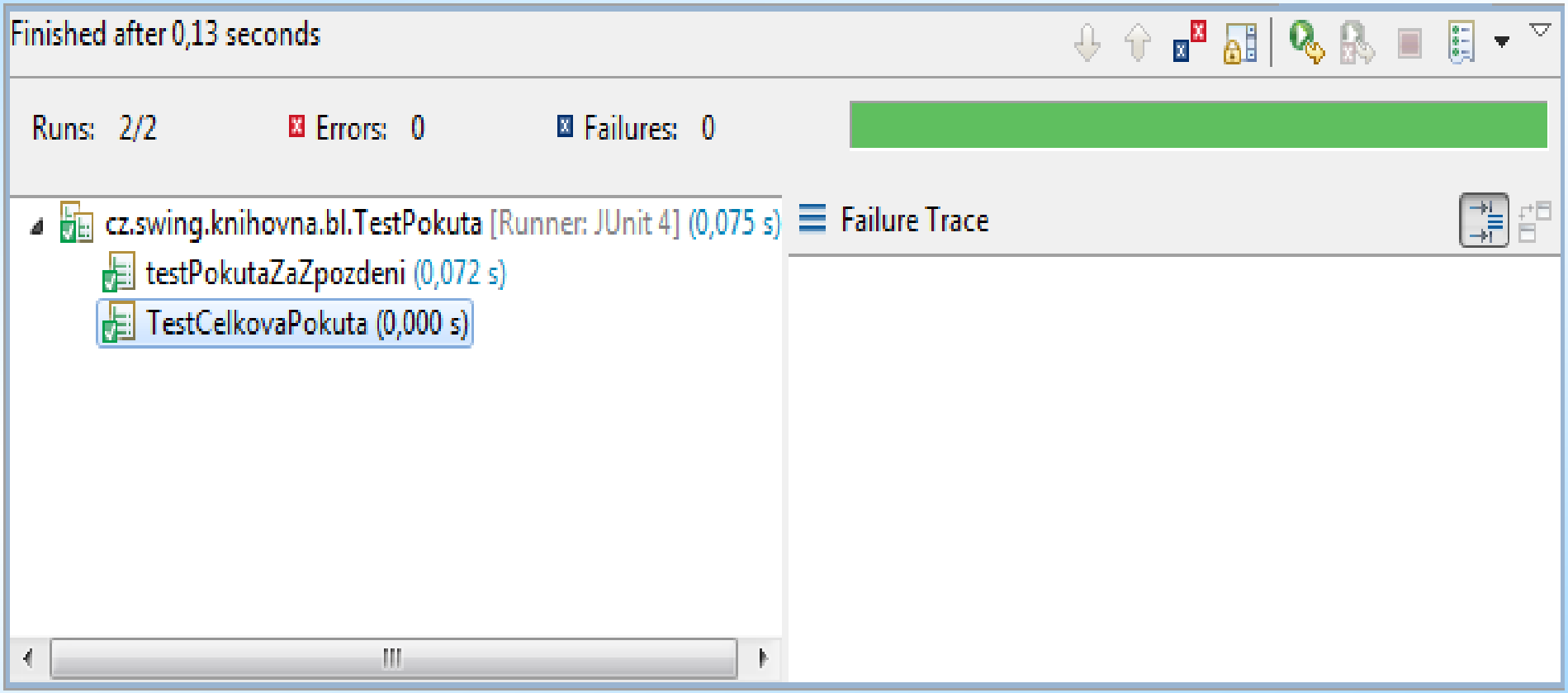
- Příprava před spuštěním testů
  - @Before
- Aktivity po skončení testů
  - @After
- Očekávání vyjímky
  - @Test(expected=IOException)

Jednotkové testy

- JUnit – vyhodnocení testu
  - Kontrola shody očekávané a skutečné hodnoty
    - assertEquals
  - Kontrola pravdivostní hodnoty
    - assertTrue, assertFalse
  - Kontrola na null hodnotu
    - assertNull, assertNotNull

Jednotkové testy

- JUnit - Eclipse



Jednotkové testy

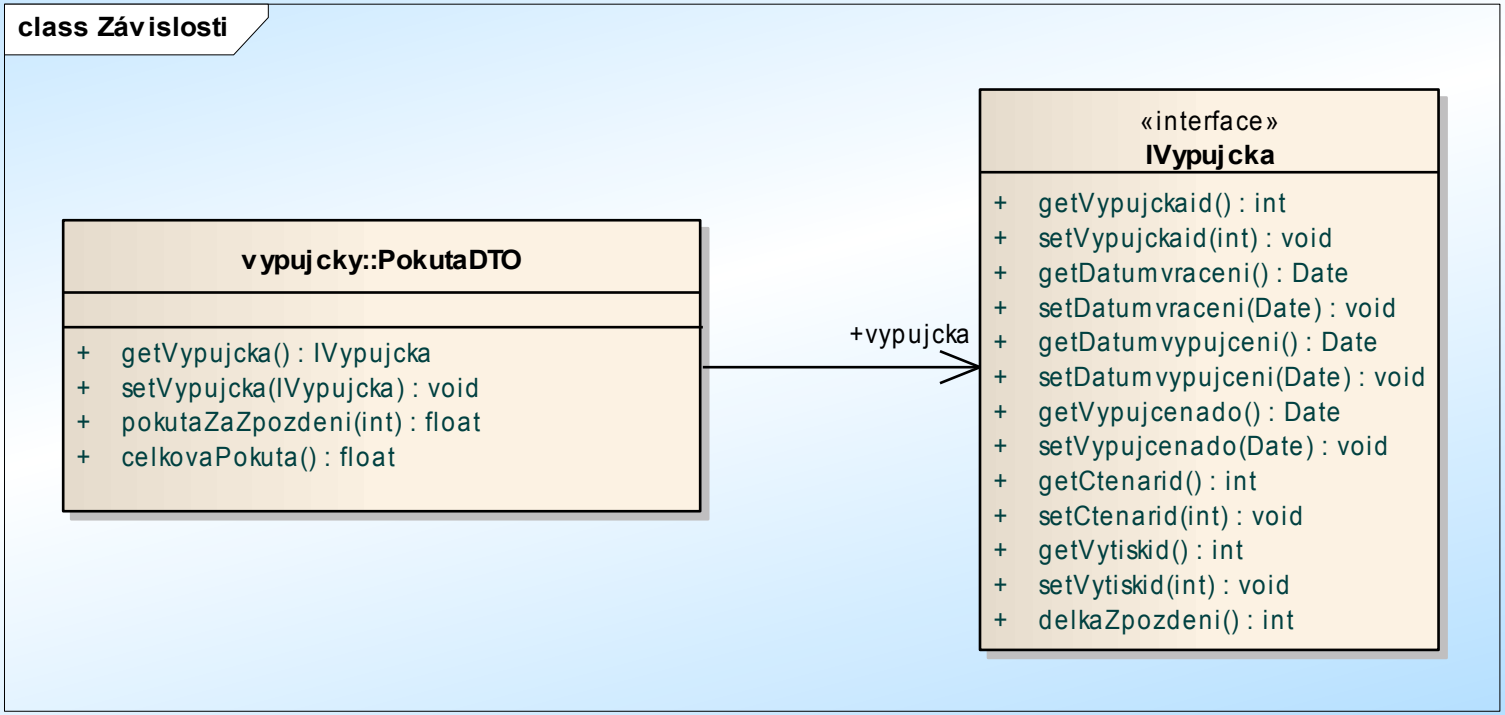
- Problém - provázanost na okolí
  - Izolované testování
  - Neexistující implementace
  - Vedlejší efekty
  - Doba provedení testu

Jednotkové testy

- Problém – provázanost na okolí
  - Použití Mock objektů
    - Ruční implementace
    - Existující nástroje
      - jMock: http://www.jmock.org/
      - EasyMock: http://easymock.org

Jednotkové testy

- Problém – provázanost na okolí



Jednotkové testy

- Problém – provázanost na okolí

```
public class PokutaDTO {
   public static final float SPRAVNI_POPLATEK=50;
   private IVypujcka vypujcka;

   ...

   public float celkovaPokuta(){
      int delkaZpozdeni = vypujcka.delkaZpozdeni();
      float pokuta = pokutaZaZpozdeni(delkaZpozdeni);
      float celkem=SPRAVNI_POPLATEK+pokuta;
      return celkem;

   }
}
```

Jednotkové testy

- jMock

```
public class TestPokuta {
   private Mockery context = new Mockery();
   private IVypujcka mockVypujcka;

   @Before
   public void nastaveni(){
      mockVypujcka=context.mock(IVypujcka.class);
      context.checking(new Expectations(){{
         oneOf (mockVypujcka).delkaZpozdeni();
           will(returnValue(9));
         oneOf (mockVypujcka).delkaZpozdeni();
           will(returnValue(10));
      }});
   }
}
```

Jednotkové testy

- jMock

```
@Test
public void TestCelkovaPokuta(){
   PokutaDTO pokuta = new PokutaDTO();
   pokuta.setVypujcka(mockVypujcka);
   float celkem = pokuta.celkovaPokuta();
   assertEquals(140,celkem);
   celkem = pokuta.celkovaPokuta();
   assertEquals(250,celkem);
}
```

Jednotkové testy

- jMock – využití
  - Určení akce na základě pořadí volání
  - Určení akce na základě vstupních parametrů
  - Určení akce na základě stavu

Jednotkové testy

Dotazy?

Testy komponent

- Testování izolovaných komponent
- Podobné jako jednotkové testy
- Testují větší části funkčností
- Zaměřeny na komunikaci mezi třídami, které reprezentují nějakou komponentu (část systému)
- Typicky - White Box

Testy komponent

- Nástroje
  - JUnit
  - NUnit
  - jMock
  - EasyMock

Testy komponent

Dotazy?

Integrační testy

- Testování spolupráce komponent
- Simulují reálné prostředí
  - Připojení k databázi
  - Probíhají ve standardním prostředí (kontejneru)
- Trvají delší dobu
- Typ
  - Black Box – integrace s komponentami jiných dodavatelů
  - White Box – integrace vlastních komponent

Integrační testy

- Připojení k databázi
  - Příprava testovacích dat
  - Testy spoléhají na stav dat v databázi
    - Průchodem testů se tento stav změní
    - Nutné obnovovat do původního stavu
    - Nastavování před každým testem velmi zpomaluje průběh testů

Integrační testy

- Obnovení dat do stavu před testy
  - Obnovou ze zakládacích skriptů
  - Výměna standardní databáze za databázi v paměti
  - Existující řešení
    - Spring

Integrační testy

- Spring
  - Řeší problém se správou dat v databázi

```
@Test
@Rollback
public void TestUlozVytisk(){
      IVytiskDAO vytiskDAO = new VytiskDAO();
      int pocetVytisku = vytiskDAO.pocetVytisku();
      Vytisk vytisk = new Vytisk();
      vytisk.setEvidencnicislo("11111");
      vytiskDAO.ulozVytisk(vytisk);
      int pocetVytiskuNovy = vytiskDAO.pocetVytisku();
      assertEquals(pocetVytisku+1,pocetVytiskuNovy);
}
```

Integrační testy

- Spring
  - Řeší problém s testováním v kontejneru

```
@ContextConfiguration(locations="services.xml")
public class TestKnihaVypujcek{
  @Autowired
  private IKnihaVypujcek knihaVypujcek;

  @Test
  ...
}
```

Integrační testy

- Problém – délka běhu
  - Řešení pomocí serverů pro kontinuální integraci
    - Automatický build
    - Automatické nasazení
    - Automatické spuštění testů
    - Reportování výsledků testů
  - Nástroje
    - Jenkins: http://jenkins-ci.org/
    - CruiseControl: http://cruisecontrol.sourceforge.net

Integrační testy

- Výhody využití serveru pro kontinuální integraci
  - Nezatěžuje lokální počítač vývojáře
  - Sledování metrik
    - Pokrytí kódu testy
    - Množství chyb
  - Otestování aplikace včetně procesu nasazení

Integrační testy

Dotazy?

Systémové testy

- Testují aplikaci jako celek
- Typicky - Black Box
- Testovací scénáře – scénáře UC modelu

Systémové testy

- Nástroje
  - Automatické testování GUI
    - Selenium: http://seleniumhq.org/
  - Automatické testování služeb
    - SoapUI: http://www.soapui.org/

Systémové testy

Dotazy?

Statická analýza kódu

- Nevyžaduje spuštění aplikace
- Založena na detekci chybových vzorů
  - Typické problémy
    - Detekce duplicit v kódu
    - Mrtvé části kódu
    - Dodržování jmenné konvence
    - Detekce možných problémů
      - Zachycení výjimky bez jejího řešení (prázdná klauzule catch)

Statická analýza kódu

- Nástroje
  - PMD: http://pmd.sourceforge.net/
  - FindBugs: http://findbugs.sourceforge.net/

Statická analýza kódu

Dotazy?

Děkuji za pozornost.