Seminář Java

Ladění

Radek Kočí

Fakulta informačních technologií VUT

Únor 2008

Ladění programu

Pro ladění programů v Javě lze využít

- kontrolní tisky: `System.err.println(...)`
- řádkový debugger `jdb`
- integrovaný debugger v IDE
- speciální nástroje na záznam běhu balíků

Uvědomte si, že žádný nástroj za nás nevymyslí, JAK máme své třídy testovat. Pouze nám pomůže ke snadnějšímu sestavení a spuštění testu.

Ladění programu

- standardní klíčové slovo (od JDK1.4) `assert`
  - `assert booleovský_výraz`
- testovací nástroje typu JUnit (a varianty – `HttpUnit`, ...)
  - metoda `assertEquals()`
  - metoda `assertTrue()`
  - ...
  - `http://junit.org/`
- pokročilé nástroje na běhovou kontrolu platnosti invariantů, vstupních, výstupních a dalších podmínek
  - např. jass (Java with ASSertions),
  - `http://csd.informatik.uni-oldenburg.de/~jass/`

Ladění programu – assert

```
public class AssertDemo {
    public static void main(String args[]) {
        int x = 10;
        boolean enabled = false;

        assert enabled = true;

        System.out.println("Assertions are " +
            (enabled ? "enabled" : "disabled");

        assert x < 0 : "x is not < 0";
    }
}
```

Ladění programu – assert

- přeložit s volbou `-source 1.4`
- spustit s volbou `-ea` (`-enableassertions`)
- dojde-li za běhu programu k porušení podmínky stanovené za `assert`, vznikne běhová chyba (`AssertionError`) a program skončí

Ladění programu – JUnit

Instalace JUnit

- stáhnout si distribuci testovacího prostředí (stačí binární) `http://junit.org`
- nainstalovat (rozbalit do adresáře) → archiv jar

Java Archive (JAR)

- archiv (zip) obsahující
  - hierarchii balíků a class soubory
  - jakékoliv jiné soubory (obrázky pro aplety apod.)
- použití
  - `javac -cp junit.jar ...`

Ladění programu – JUnit

Postup (starší verze)

- napsat testovací třídu (třídy) – obvykle rozšiřují (dědí) třídu `junit.framework.TestCase`
- testovací třída obsahuje metody
  - metodu pro nastavení testu – `setUp()`
  - testovací metody – `test`*`Neco`*`()`
  - úklidovou metodu – `tearDown()`
- testovací třídu spustit v textovém nebo grafickém prostředí
  - `junit.textui.TestRunner`
  - `junit.swingui.TestRunner`
- testování zobrazí, které testovací metody případně selhaly

Ukázka (starší verze)

```
public class JUnitDemo extends TestCase {
   Zlomek x, y, z;

   public void setUp() {
      x = new Zlomek(2,3);
      y = new Zlomek(4,6);
      z = new Zlomek(4,3);
   }
   public void testRovna() {
      assertEquals("2/3 = 4/6.", x, y);
   }
   public void testSoucet() {
      Zlomek z = x.plus(y);
      assertEquals("2/3 + 4/6 = 4/3.", z, soucet);
   }
}
```

Ladění programu – JUnit

Využití anotací (annotation)

- anotace definují dodatečné informace a data o programu bez přímého vlivu na program
- využití anotací
  - při kompilaci – detekce chyb, možnost generování dalšího kódu, ...
  - za běhu – nástroje a knihovny mohou na základě anotace přizpůsobit sémantiku

Static import

Přístup ke statickým členům

- `double r = Math.cos(Math.PI * theta);`
- Využití statického importu

```
 import static java.lang.Math.PI;
//import static java.lang.Math.*;
...

double r = cos(PI * theta);
```

- používat velice opatrně! (kolize identifikátorů, těžko čitelný kód, ...)

Ladění programu – JUnit

Ukázka využití anotací (annotation)

```
import org.junit.*;
import static org.junit.Assert.*;

public class TestHW {
    @Test public void xxx() {
        ...
        assertTrue(x>10);
    }
```

```
java org.junit.runner.JUnitCore homework1.TestHW
```