WPA - Ladění a testování webových aplikací

Martin Ledvinka

21. října 2015

Obsah

NetBeans + Maven tips

- **Auto-deploy**
  - *Properties -¿ Run -¿ Deploy on Save*
- **Spuštění Maven cílů**
  - *Custom -¿ Goals...*
- Maven přepínače
  - -X
  - -DskipTests=true

Různé úrovně testování

Existují různé úrovně testování:

- Jednotkové (unit),
- Integrační,
- Funkční.

Další typy testování

Dále se testování dá dělit na mnoho skupin (ne nutně disjunktních):

- Black-box a White-box,
- Regresní,
- Akceptační,
- Zátěžové,
- Usability.

Přehled frameworků

- JUnit,
- TestNG.

Základní API

Anotace

- `@Test`
- `@Before`
  - `@BeforeMethod` *TestNG*
  - Před spuštením každého testu
- `@After`
  - `@AfterMethod` *TestNG*
  - Po doběhnutí každého testu (bez ohledu na výsledek)
- `@BeforeClass`
  - Metoda musí být `static`
  - Spustí se jen jednou, před spuštěním prvního testu
- `@AfterClass`
  - Metoda musí být `static`
  - Spustí se jen jednou, po doběhnutí poslední testu (a po @After)

Základní API

Metody

- `Assert.assertTrue(expr)`, `Assert.assertFalse(expr)`
  - `expr` musí být `true`, resp. `false`
- `Assert.assertEquals(expected, actual)`
  - Obě hodnoty se musí rovnat (při použití `equals`)
- `Assert.assertSame(expected, actual)`
  - Obě hodnoty odkazují na stejný objekt (použije se ==)
- `Assert.assertNull(expr)`
  - `expr` musí být `null`
- `Assert.assertNotNull(expr)`
  - `expr` nesmí být `null`
- `Assert.fail(message)`
  - Okamžitý fail testu

Unit test by měl

- Testovat jen jednu věc (funkcionalitu),
- Testovat:
  - Očekávané hodnoty vstupních parametrů,
  - Nevalidní hodnoty,
  - Mezní hodnoty,
- Být nezávislý na ostatních testech.

Code Coverage

- Měříme pokrytí kódu testy,
- Pokrytí tříd, řádků, větví,
- **100% pokrytí neznamená bezchybný kód!!!**,
- JaCoCo, Cobertura.

Logování

- Ladící/informační výpisy,
- Zalogování výjimek,
- Informace o běhu aplikace,
- Nastavitelná úroveň logování – podrobnější výpisy pří vývoji, jen základní info v produkci.

Přehled frameworků

- JUL (java.util.logging)
  - součástí JDK, nejsou nutné žádné další knihovny,
  - mnoho logovacích úrovní,
- Apache Log4j
  - Po dlouhou dobu lídr mezi logovacími nástroji,
  - Snadná konfigurace,
- Simple Logging Facade for Java (SLF4J)
  - Logovací interface, vyžaduje ještě implementaci,
  - Rozdělení je výhodné – možno přepínat mezi JUL, Log4j, logback,
- logback
  - Nástupce původního Log4j.

Literatura

- JaCoCo Maven plugin

  `http://eclemma.org/jacoco/trunk/doc/maven.html`
- Předmět BI-ATS na FIT ČVUT

  `https://edux.fit.cvut.cz/courses/BI-ATS/`
- Logback manual

  `http://logback.qos.ch/manual/`