JUnit – testowanie aplikacji

Paweł Chodarcewicz

Plan prezentacji

- Droga do JUnit
- Krótkie omówienie JUnit
- Paradygmat „programowanie sterowane testami”
- Pokaz funkcjonalności JUnit
- JUnit w wersji 4, co nowego?
- Odnośniki

Co to jest JUnit?

JUnit to biblioteka wspomagająca pisanie testów i wielokrotne, automatyczne ich uruchamianie.

Kod JUnit jest ogólnodostępny.

Pierwsza wersja JUnit została napisana przez Ericha Gamma i Kenta Beck (2000).

Obecnie popularne są biblioteki XUnit. JUnit można podłączyć do większości IDE

Testowanie bez dodatkowych narzędzi

- Wypisywanie ogromnych logów
  - dużo różnych informacji;
  - trudno pamiętać, jaki wynik być powinien
- Asercje to za mało
- Używanie debugera, aby prześledzić działanie programu
- Testowanie przez człowieka

JUnit w wielkim skrócie

Etapy pracy z JUnit:

- Pisanie testu (interfejs TestCase). Często korzysta się z asercji.
- Grupowanie testów w grupy (interfejs TestCase)
- Uruchamianie testów (różne mechanizmy uruchamiania: graficzne [zintegorwane z IDE], linia poleceń)

Prosty przykład działania JUnit

Pokaz „na żywo” w środowisku Eclipse

JUnit i programowanie obiektowe

JUnit idealnie wpasowuje się w koncepcje programowania obiektowego (testowanie „kontraktu”)

JUnit a testowanie API

JUnit – nazwa wiąże się z testowaniem „Unitów”

Programowanie sterowane testami (ang. Test Driven Development)

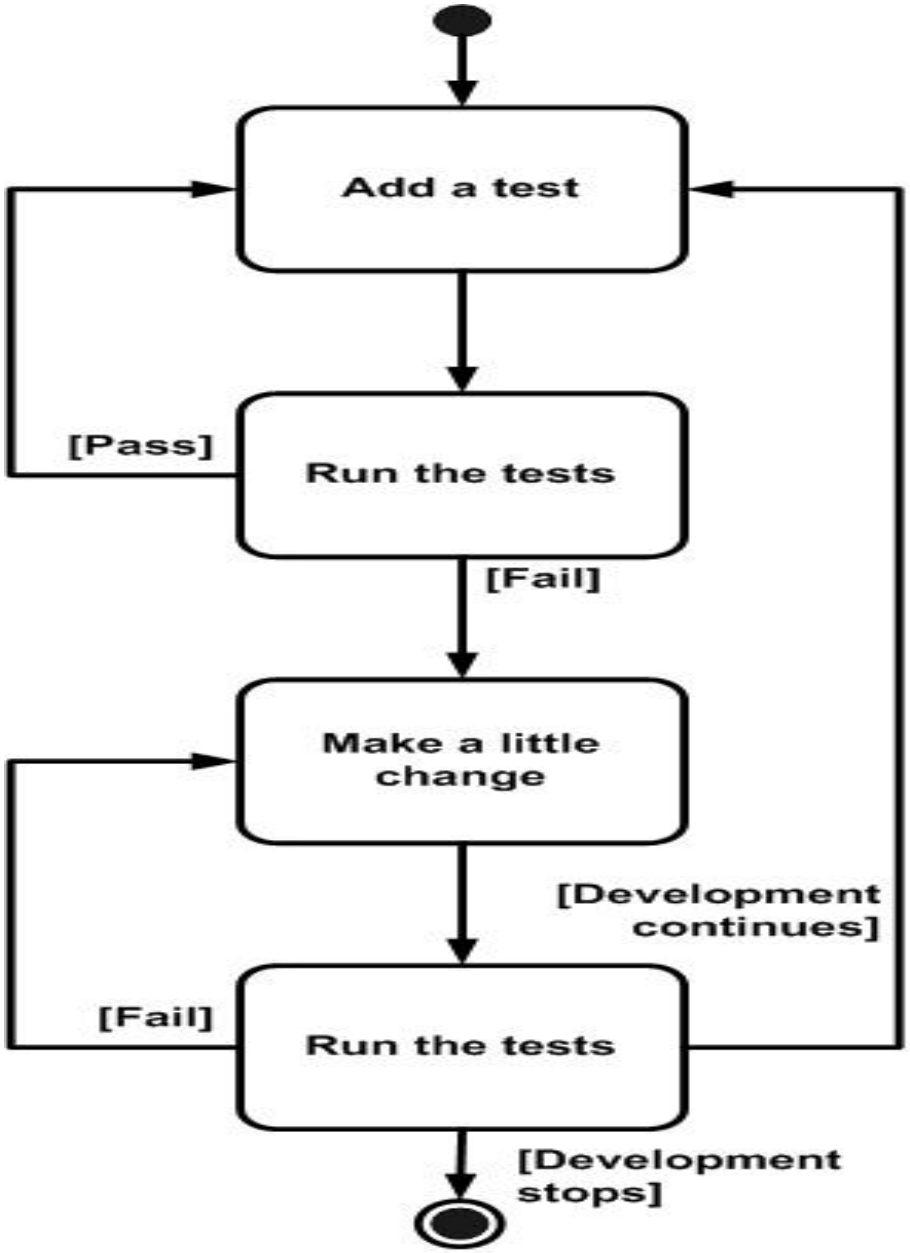
- Testy powstają **przed** kodem
- Pisanie kodu, ale tylko tyle, aby testy przeszły
- **Refaktoryzacja** – usuwanie zduplikowanego kodu
- **Każdy** fragment kodu musi być objęty jakimiś testami
- Przed usunięciem odkrytego błędu pisze się test wykrywający ten błąd

Schemat

Test

Driven

Development



Zalety Test Driven Development

- Zmusza do skupienia się na projektowaniu (a nie od razu przejście do pisania kodu)
- Pozwala skorzystać z tworzonego API przed jego powstaniem – praktyczny test, czy projekt jest dobry
- Błędy są wykrywane dużo wcześniej niż przy standardowej metodologii
- Zbiór testów jest rodzajem dokumentacji – wciąż aktualizowanej!!!!!!!

JUnit w praktyce

Pokaz „na żywo” w środowisku Eclipse.

Nowości w JUnit 4.0

- Oparty o adnotacje (dużo wygodniej niż dotąd)
- Stworzony dla Java 5
- Wyeliminowanie „test runnera” – okienkowy interfejs do wyników testów jedynie przez IDE
- Specyfikacja ukończona, implementacja jeszcze nie

Nowości w JUnit 4.0

```
import org.junit.Test;
import junit.framework.TestCase;

public class AdditionTest extends TestCase {
  private int x = 1;
  private int y = 1;

//adnotacja decyduje, czy coś jest metodą testową a nie prefix „test”
  @Test public void addition() {
    int z = x + y;
    assertEquals(2, z);
  }
}
```

Nowoście w JUnit 4.0

```
import org.junit.Assert;

    /* nie potrzeba dziedziczyć z TestCase – wystarczy adnotacja, *
     * którą  metodę uruchomić w czasie testowania                */
public class AdditionTest {
  private int x = 1;
  private int y = 1;

  @Test public void addition() {
    int z = x + y;
    Assert.assertEquals(2, z);
  }
}
```

Nowości w JUnit 4.0

//zamiast setUp() adnotacja @Before (może być wielokrotna)

```
@Before protected void findTestDataDirectory() {
    inputDir = new File("data");
    inputDir = new File(inputDir, "xslt");
    inputDir = new File(inputDir, "input");
}

 @Before protected void redirectStderr() {
    System.setErr(new PrintStream(new
     ByteArrayOutputStream()));
}
```

Nowości w JUnit 4.0

//zamiast metoty tearDown() adnotacja @After (może być wielokrotna)

```
@After protected void disposeDocument() {
  doc = null;
  System.gc();
}
```

Nowoście w JUnit 4.0

```
// „klasowe” setUp() i tearDown()

@BeforeClass protected void connect() {
    DB.connect();
}

@AfterClass protected void disconnect() {
    DB.disconnect();
}
```

Nowości w JUnit 4.0

```
/*
Dużo łatwiejsze wyrażenie, że spodziewa się
   (bądź nie) wyjątku
*/
@Test(expected=ArithmeticException.class)
  public void divideByZero() {
    int n = 2 / 0;
}
```

Nowości w JUnit 4.0

// możliwość kontroli długości działania metody

```
@Test(timeout=2000)
 public void sort(){
    myArray.sort();
    myArray.assertSorted();
}
```

Odnośniki

- JUnit (http://www.junit.org)
- JUnit 4.0 (http://www-128.ibm.com/developerworks/java/library/j-junit4.html)
- XP – przykład Test Driven Development (http://www.xprogramming.com/)