JUnit-pikaohje NetBeans-käyttäjille

Testataan kokonaislukupinon toteuttavaa luokkaa Stack, jolla on seuraavat metodit:

```
void push(int data)
int pop () throws IllegalStateException;
boolean empty()
```

pop heittää poikkeuksen jos operaatio suoritetaan tyhjälle pinolle.

Oletetaan, että luokka on pakkauksessa stack.

JUnit-testitapaukset luodaan seuraavasti:

1. Klikkaa hiiren oikealla napilla projektia ja valitse *New -> Other.. -> JUnit (Categories) -> JUnit Test*
2. Anna testiluokalle nimi, esim. "StackTest" (huomaa, että luokan nimen lopussa on *oltava* Test)
    a. Jos projektisi on packagessa (ei siis *default package*), niin luo saman niminen pakkaus Test packages -kohtaan ja valitse se myös testiluokalle.
    b. esimerkissämme pakkaus on siis stack
3. Valitse versio JUnit 4.x
4. *Test Packages* -kansioon ilmestyy nyt *StackTest.java*
5. Testit ajetaan valitsemalla *Run*-valikosta *Test "Projektinnimi"*

Syntyvä testiluokka näyttää seuraavalta:

```
// alussa importeja...

public class StackTest {

      public StackTest() {
      }

      @BeforeClass
      public static void setUpClass() throws Exception {
      }

      @AfterClass
      public static void tearDownClass() throws Exception {
      }

      @Before
      public void setUp() {
      }

      @After
      public void tearDown() {
      }
}
```

Aluksi valmiista metodeista ei ole oikeastaan muilla merkitystä paitsi setUp():illa. Se on metodi, joka suoritetaan ennen jokaista testiä. Testit lisätään luokan sisään jokainen omaksi metodikseen. Metodin alkuun (tai yläpuolelle) tulee annotaatio @Test

Luokan Stack testit, mukana selittäviä kommentteja:

```
package stack;

import org.junit.Before;
import org.junit.Test;
import static org.junit.Assert.*;

public class StackTest {
```

määritellään pinoviite luokkamuuttujaksi, jotta kaikki testit pääsevät pinoon käsiksi

```
	Stack pino;
```

setUp()-metodi suoritetaan ennen jokaista testiä, eli jokaista testiä varten luodaan uusi pino

```
	@Before
	public void setUp() {
		pino = new Stack()
	}
```

sitten varsinaiset testit, jotka siis ovat kukin luokan metodeja， ensin testataan että pino on alussa tyhja ennen metodia on siis suoritettu setUp()-metodi

```
	@Test
	public void alussaTyhja() {
		assertTrue(pino.empty());
	}
```

assertTrue() siis “vaatii”, että sen testaaman ehdon totuusarvo on true, jos näin ei ole, testi ei mene läpi

seuraava testi testaa että pino on pino, eli alkiot tulevat ulos käänteisessä pinoonlaittojärjestyksessä ja että kun kaikki alkiot on poistettu, on pino taas tyhjä

```
	@Test
	public void alkiotTulevatOikeassaJarjestyksessaPinosta() {
		pino.push(1);
		pino.push(2);
		pino.push(3);
		assertEquals(3, pino.pop());
		assertEquals(2, pino.pop());
		assertEquals(1, pino.pop());
		assertTrue(pino.empty());
	}
```

huomaa edellisessä vertailu assertEquals(), vertailuja on monia muitakin

testataan että tyhjennetty pino toimii kuten uusi

```
	@Test
	public void tyhjennettyPinoToimiiKutenUusi() {
		pino.push(1);
		pino.push(2);
		pino.pop();
		pino.pop();
		pino.push(3);
		assertEquals(3, pino.pop());
		assertTrue(pino.empty());
	}
```

tyhjästä poppaamisen pitää aiheuttaa poikkeus, asia testataan siten, että kutsutaan virheen ilmoittavaa metodia fail() jos poikkeus ei ole syntynyt

```
	@Test
	public void tyhjastaPinostaPoppaaminenAiheuttaaPoikkeuksen(){
		try{
			pino.pop();
			// jos tullaan tänne on tapahtunut virhe
				fail();
			} catch( IllegalStateException e){
				// on tarkoitus päätyä tänne
			}
	}
}
```

Kannattaa huomioida testitapausten nimeäminen, metodien nimet ovat pitkiä, mutta kuvaavat tarkasti mitä metodi tekee.

Testitapauksilla voi usein korvata main-metodin käytön kokonaan. Testitapauksethan toimivat kuten main. Näin testatun pinoluokan voi sitten turvallisesti kytkeä sitä käyttävään ohjelmaan.

Testikattavuus

NetBeansiin on helppo asentaa plugin jonka avulla voidaan mitata testien kattavuus. Kattavuudella tarkoitetaan tässä yhteydessä ns. rivikattavuutta, eli mitä kaikkia rivejä on suoritettu testitapauksia ajettaessa. Kattavuuden pitäisi tietysti olla lähellä 100 prosenttia. Ohje testikattavuuspluginin asentamiseen ja käyttöön:

http://wiki.netbeans.org/TutorialCodeCoveragePlugin