Workshop Unit Testing & JUnit

Kuliah Software Quality Assurance

1. Pastikan bahwa program java dan javac sudah terinstall di komputer masing-masing dengan cara membuka command line (windows) atau terminal (linux) dan mengetikkan perintah:

   ```
   java --version
   javac --version
   ```

2. Unduh berkas `JUnit`, `Hamcrest` dan `MySet.java` di Elen STT-NF.
3. Untuk mengurangi redudansi dalam dokumen tutorial ini, berikut adalah perintah-perintah yang akan sering dijalankan:

   ```
   //compile MySet.java
   javac MySet.java

   //compile MySetTest.java
   javac -cp .;junit-4.12.jar MySetTest.java

   //menjalankan unit testing
   java -cp .;junit-4.12.jar;hamcrest-core-1.3.jar
   org.junit.runner.JUnitCore MySetTest
   ```

4. Pada tutorial kali ini kita akan melakukan unit testing pada file MyTest.java dengan menggunakan JUnit. Berikut ini adalah spesifikasi dari MySet.java:
   a. Dapat menambahkan element.
   b. Dapat menghapus element.
   c. Dapat mencari element.
   d. Tidak ada duplikasi element.
5. Buatlah sebuah berkas baru dengan nama MySetTest.java
6. Tulis kode berikut ini pada file MySetTest.java

   ```
   import static org.junit.Assert.assertEquals;
   import org.junit.Test;
   ```

```
public class MySetTest
{
   @Test
    public void testAddElement()
    {
        MySet set = new MySet();
        set.add("Budi");
        assertEquals(1, set.size());
    }
}
```

7. Kode diatas dibuat dengan tujuan untuk memvalidasi apakah requirement “Dapat menambahkan element” dengan cara melakukan testing pada method “add” dan mengecek apakah size dari MyTest akan bertambah seiring penambahan element. Jika diperhatikan pada potongan kode diatas, terdapat sebuah anotasi “@Test” yang menandakan bahwa method yang diberi anotasi tersebut merupakan test case untuk JUnit.

8. Compile berkas MySet.java dengan menggunakan perintah:

```
javac MySet.java
```

9. Compile berkas MySetTest.java dengan menggunakan perintah:

```
javac -cp .;junit-4.12.jar MySetTest.java
```

10. Lalu jalankan JUnit dengan menggunakan perintah:

```
java -cp .;junit-4.12.jar;hamcrest-core-1.3.jar
org.junit.runner.JUnitCore MySetTest
```

11. Output yang dihasilkan adalah sebagai berikut:

```
JUnit version 4.12
.
Time: 0.021

OK (1 test)
```

12. Output diatas menandakan bahwa unit testing yang kita berhasil dijalankan.
13. Berikutnya kita akan memvalidasi "Dapat menghapus element" dengan mengetikkan kode berikut di berkas **MySetTest.java**:

```
@Test
public void testRemoveElement()
{
    MySet set = new MySet();
    set.add("Budi");
    set.add("Roland");
    set.add("Arya");

    set.remove("Roland");
    int index = set.indexOf("Roland");
    int size = set.size();

    assertEquals(-1, index);
    assertEquals(2, size);

    int index2 =  set.indexOf("Arya");
    assertEquals(1, index2);
}
```

14. Compile **MySet.java, MySetTest.java** dan kemudian jalankan JUnit (lihat point 2)
15. Potongan kode diatas akan menghasilkan output sebagai berikut:

```
JUnit version 4.12
..
Time: 0.006

OK (2 tests)
```

16. Langkah selanjutnya adalah membuat test case untuk memvalidasi "Tidak ada duplikasi element". Berikut ini adalah kode unit testing yang akan kita tambahkan di berkan MySetTest.java:

```
@Test
public void testSetContainNoDuplicateElement()
{
    MySet set = new MySet();
    set.add("Budi");
    set.add("Roland");
    set.add("Roland");
    assertEquals(2, set.size());
}
```

17. Compile berkas **MySetTest.java** dan jalankan JUnit (lihat point 2).

Tantangan:

Buatlah sebuah test case menggunakan JUnit untuk menguji requirement “Dapat mencari element”. Hint: Pastikan method indexOf() pada MySet.java mengembalikan index yang sesuai dengan lokasi element dan -1 apabila element tidak di temukan (lihat source code MySet.java).

--o0o Selamat Mencoba o0o--