Java Essentials

Unittesten met JUnit

DE HOGESCHOOL
MET HET NETWERK

Hogeschool PXL – Elfde-Liniestraat 24 – B-3500 Hasselt
www.pxl.be - www.pxl.be/facebook

INHOUD

- Wat is unittesten?
- Waarom schrijf ik unit testen?
- Wat is JUnit?
- Een voorbeeld: testen uitvoeren

Wat is unittesten?

- Unittesten is een methode om softwaremodulen of stukjes broncode (**units**) afzonderlijk te **testen**. Bij unittesten zal voor iedere **unit** een of meerdere **tests** ontwikkeld worden.
- Een **unit** is voor ons een methode in een klasse.

Waarom schrijf ik unit testen?

- Minder fouten in je code
- Je durft je code aan te passen en beter leesbaar te maken.
- Je denkt meer na over je klasse en de implementatie van de methoden.
- Bron: http://www.onjava.com/pub/a/onjava/2003/04/02/javaxpckbk.html

Wat is JUnit?

- JUnit is een open source framework ontwikkeld om unit testen te schrijven en uit te voeren in Java.

Een voorbeeld

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

public class PersoonTest {
    @Test
    public void testGetVolledigeNaam( ) {
        Persoon p = new Persoon("Flater", "Guust");
        assertEquals("Guust Flater", p.getVolledigeNaam( ));
    }

    @Test
    public void testGetVolledigeNaamIndienNaamNull() {
        Persoon p = new Persoon(null, "Guust");
        assertEquals("Guust ?", p.getVolledigeNaam( ));
    }

    @Test
   public void testGetVolledigeNaamIndienVoornaamNull() {
        Persoon p = new Persoon("Flater", null);
       assertEquals("? Flater", p.getVolledigeNaam( ));
    }
}
```

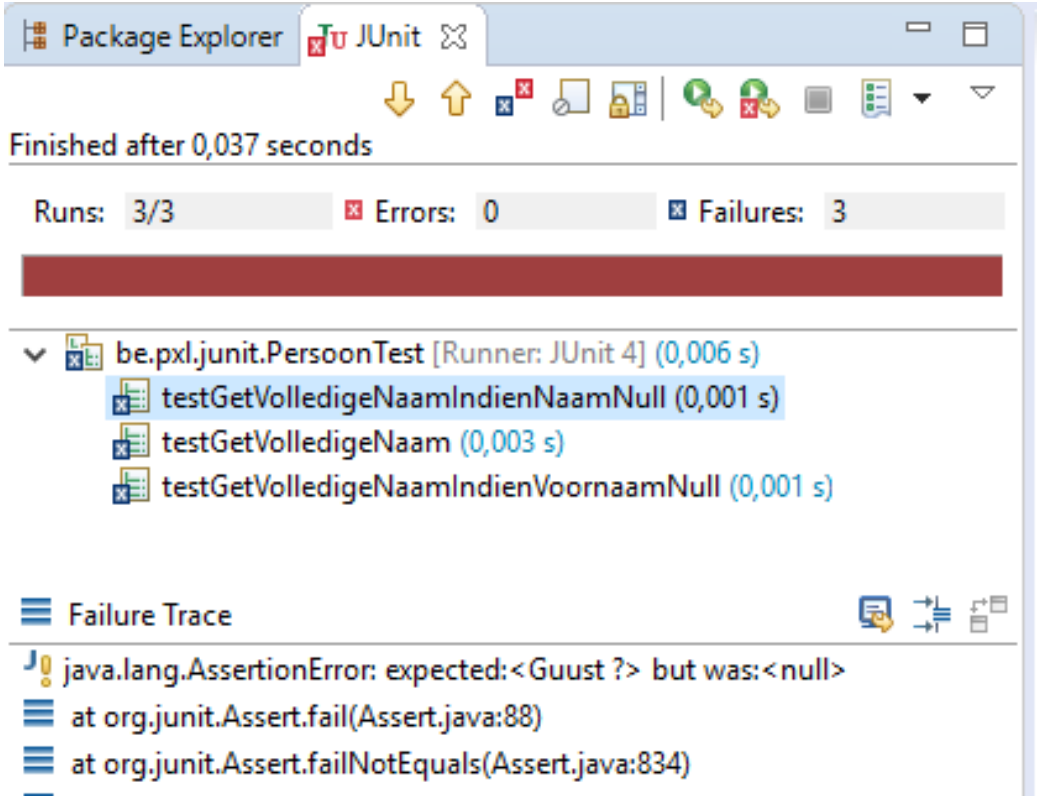
Testen uitvoeren

```
public class Persoon {
    private String naam;
    private String voornaam;

    public Persoon(String naam, String voornaam) {
        this.naam = naam;
        this.voornaam = voornaam;
    }

    public String getVolledigeNaam() {
       return null;
    }
}
```

Testen uitvoeren



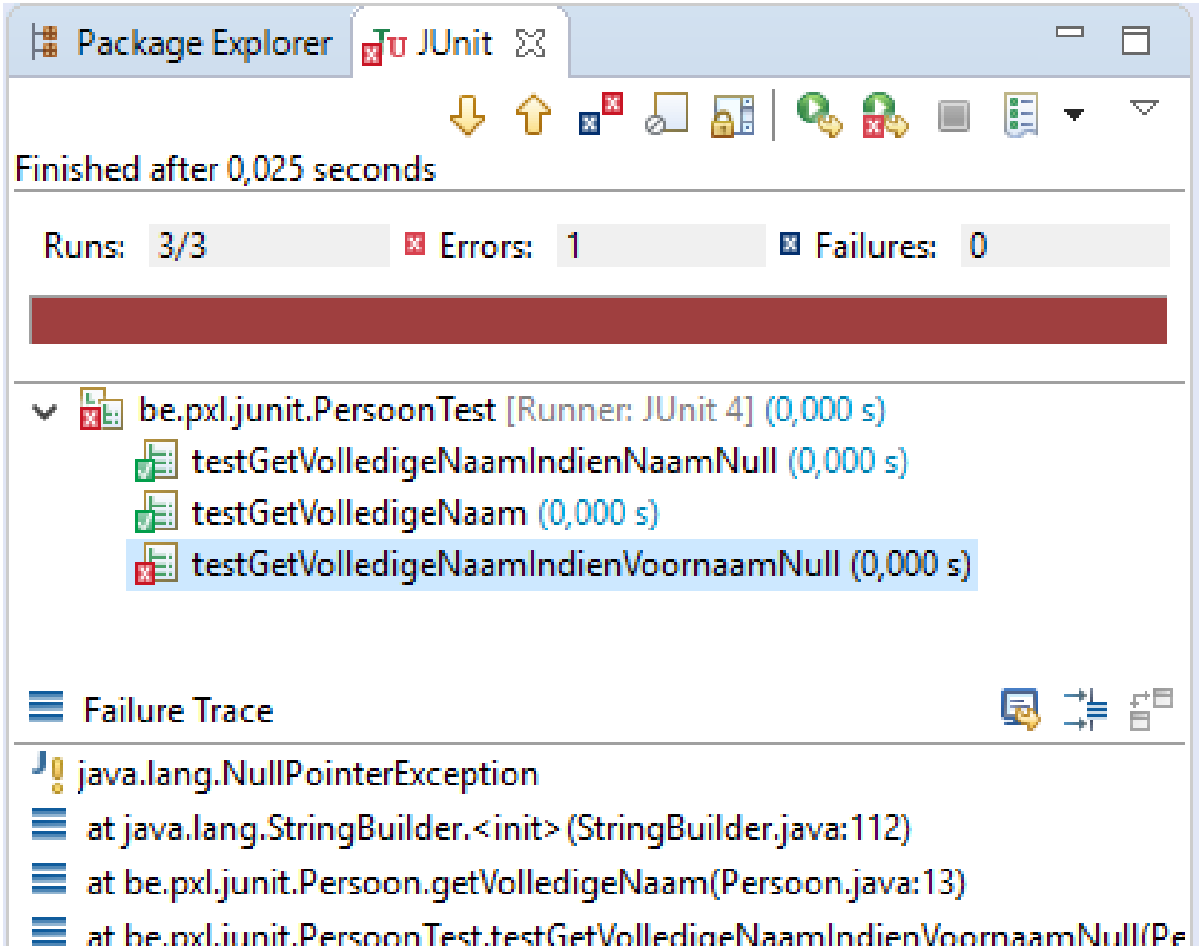
Testen uitvoeren

```
public class Persoon {
    private String naam;
    private String voornaam;

    public Persoon(String naam, String voornaam) {
        this.naam = naam;
        this.voornaam = voornaam;
    }

    public String getVolledigeNaam() {
       StringBuilder volledigeNaam = new StringBuilder(voornaam);
       if (naam == null) {
              volledigeNaam.append(" ?");
       } else {
              volledigeNaam.append(" ").append(naam);
       }
       return volledigeNaam.toString();
    }
}
```

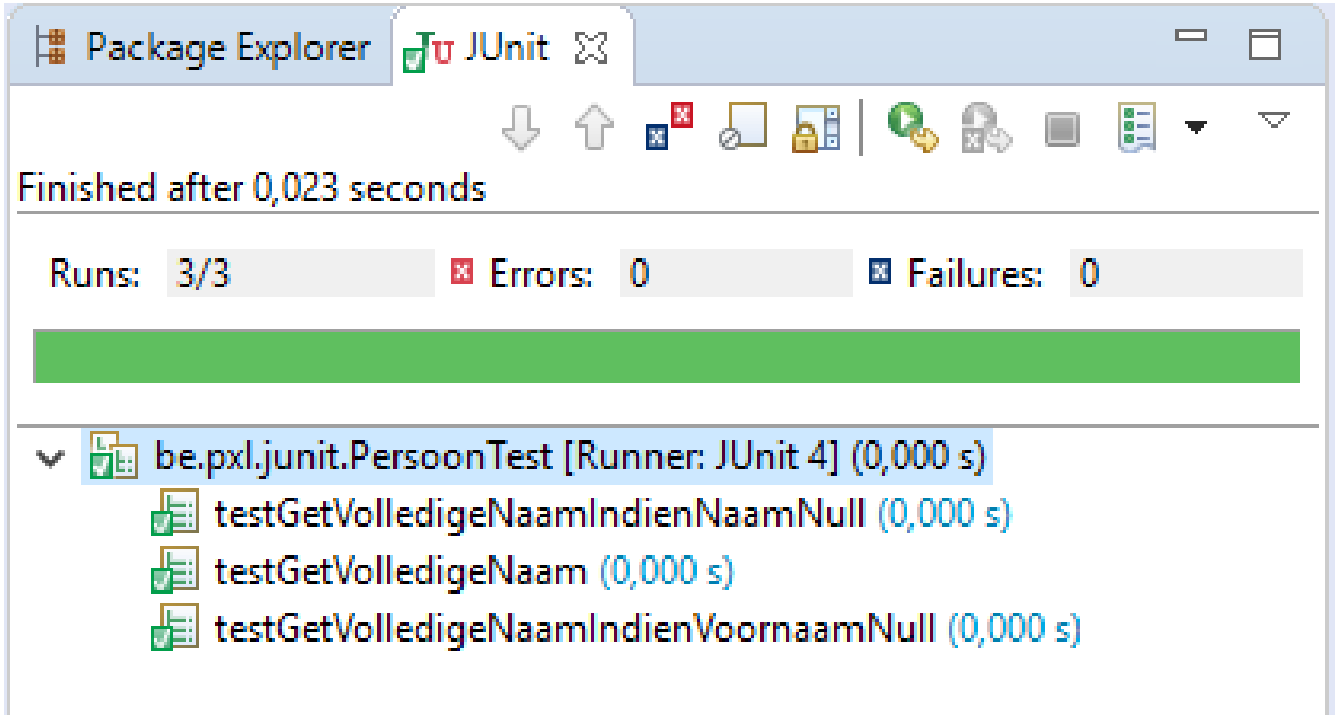
Testen uitvoeren



Testen uitvoeren

```
public String getVolledigeNaam() {
    StringBuilder volledigeNaam = new StringBuilder();
    if (voornaam == null) {
        volledigeNaam.append("?");
    } else {
        volledigeNaam.append(voornaam);
    }
    if (naam == null) {
        volledigeNaam.append(" ?");
    } else {
        volledigeNaam.append(" ").append(naam);
    }
    return volledigeNaam.toString();
}
```

Testen uitvoeren

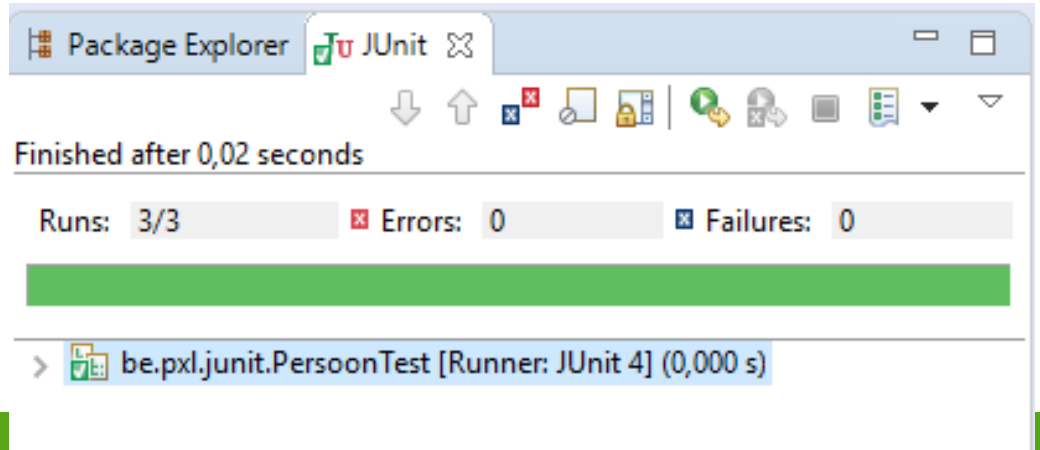


Refactor Kan je de leesbaarheid verbeteren?

Kan je dubbele code vermijden?

```
public String getVolledigeNaam() {
    StringBuilder volledigeNaam = new StringBuilder();
    volledigeNaam.append(vraagtekenIndienNull(voornaam));
    volledigeNaam.append(" ");
    volledigeNaam.append(vraagtekenIndienNull(naam));
    return volledigeNaam.toString();
}

private String vraagtekenIndienNull(String naam) {
    if (naam == null) {
        return "?";
    }
    return naam;
}
```



Methoden in org.junit.Assert

Methode	Betekenis
assertEquals()	Evalueert de gelijkheid van 2 waarden. De test slaagt als beide waarden gelijk (equal) zijn.
assertFalse()	Evaluatie van een booleaanse uitdrukking. De test slaagt indien de uitdrukking false is.
assertTrue()	Evaluatie van een booleaanse uitdrukking. De test slaagt indien de uitdrukking true is.
assertNotNull( )	Vergelijkt een object referentie met null. De test slaagt indien de object referentie niet null is.
assertNull( )	Vergelijkt een object referentie met null. De test slaagt indien de object referentie null is.
assertSame( )	Vergelijkt het geheugenadres van twee object referenties (gebruik maken van == operator). De test slaagt indien beide object referenties naar hetzelfde object verwijzen.
fail()	Zorgt ervoor de de huidige test zal fallen. Wordt regelmatig gebruikt bij het testen van exception handling.

En nu zelf aan de slag…