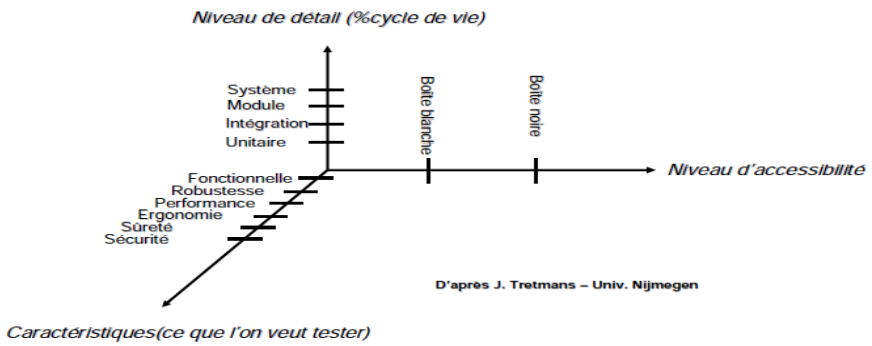
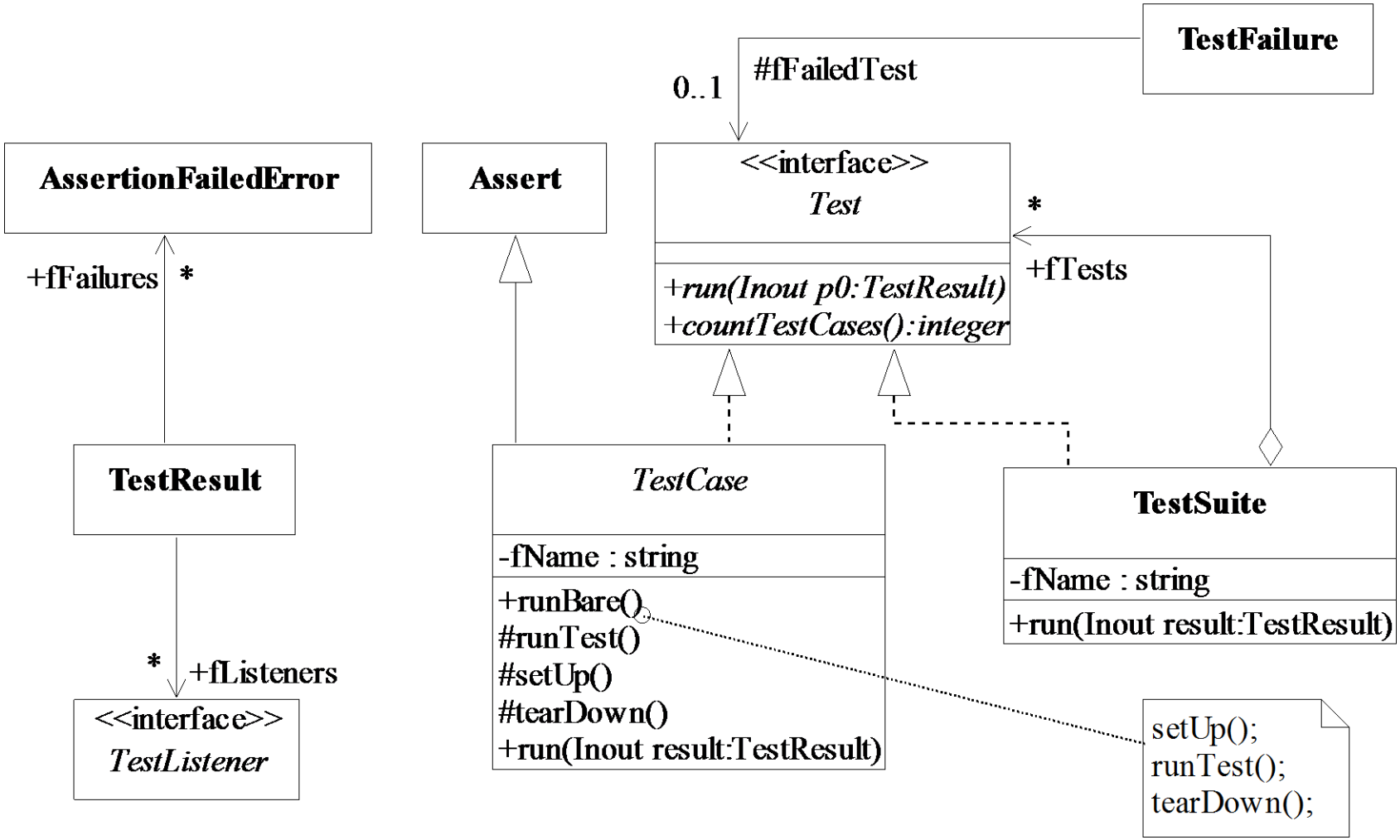
Test logiciels
Conception, Code , TDD

Brahim HAMID

brahim.hamid@irit.fr
brahim.hamid@univ-tlse2.fr

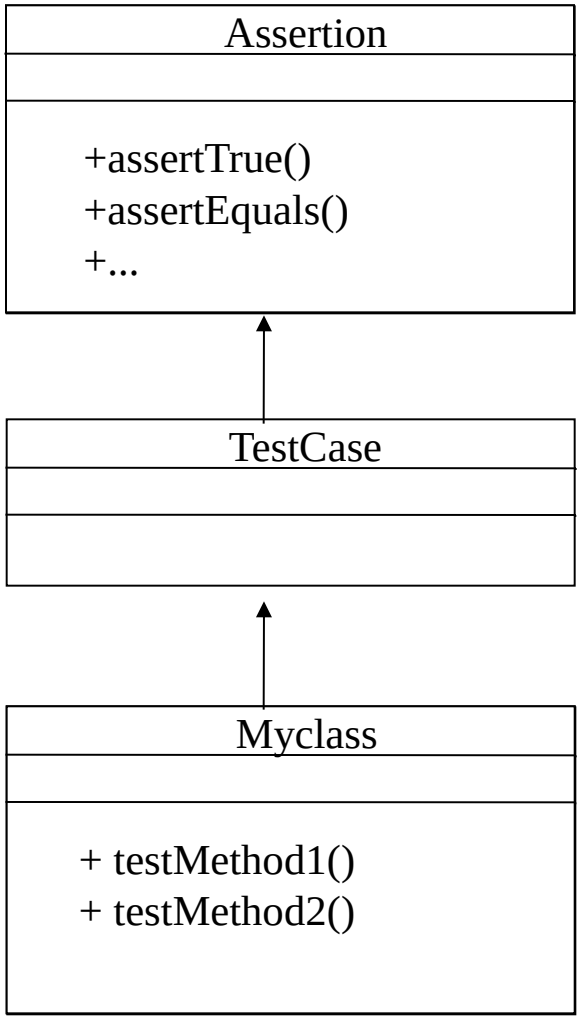


Le framework Junit



Junit : Principe

- **Une classe de tests unitaires est ssociée à une autre classe**
- **Une classe de tests unitaires hérite de la classe *junit.framework.TestCase* pour bénéficier de ses méthodes de tests**



Junit : Principe

- **Les méthodes de tests sont identifiées par des annotations Java**
- **Méthodes de test :**
  - ✓ **visibilité public, type de retour void**
  - ✓ **pas de paramètre, peut lever une exception**
  - ✓ **annotée @Test**
  - ✓ **annotation @Ignore pour ignorer un test**
  - ✓ **utilise des instructions de test**

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

public class TestSuite {
@Test
    public void testSomething() {
    // test
    }
}
```

Junit : Méthodes de test

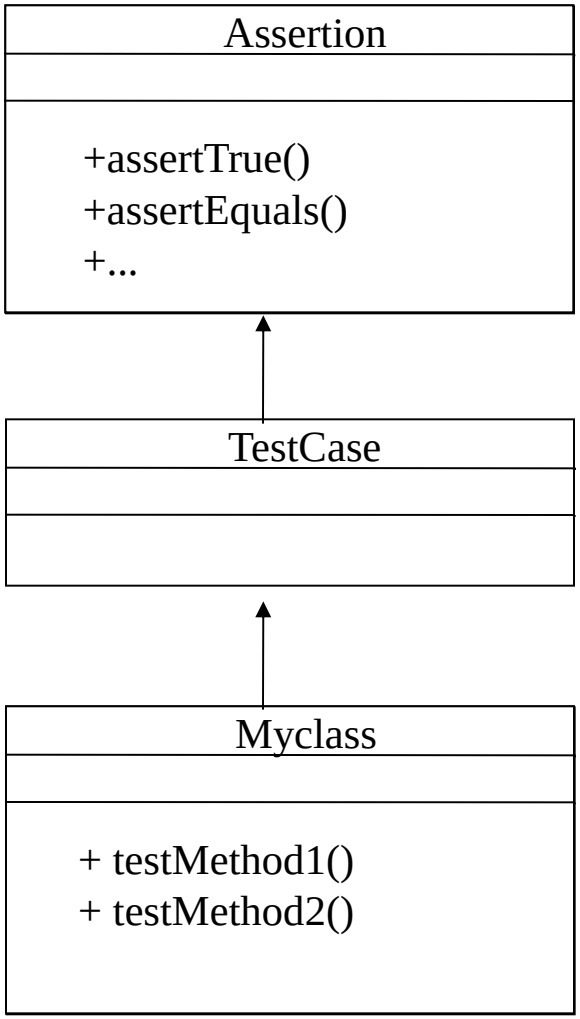
- **JUnit 3.8**
  - **Package junit.framework.* ;**
  - **public class ExempleTest extends TestCase**
- **JUnit 4**
  - **Pas de classe `a ´ étendre**

→ Code couleur Junit

- ✓ **R (Red): écrire un code de test et les faire échouer**
- ✓ **G (Green) : écrire le code métier qui valide le test**
- ✓ **R (Refactor) : remanier le code afin d'en améliorer la qualité**

→ Résultat d'un passage de test

- ✓ **Failure = erreur du test (détection d'une erreur dans le code testé)**
- ✓ **Error = erreur/exception dans l'environnement du test (détection d'une erreur dans le code du test)**



Assertions

- **Méthodes statiques déclarées dans la classe *org.junit.Assert***

(https://junit.org/junit4/javadoc/latest/)

  - **fail(String) : fait échouer la méthode de test**
  - **assertTrue(true) : toujours vrai**
  - **assertsEquals(expected, actual) : teste si les valeurs sont les mêmes**
  - **assertsEquals(expected, actual,tolerance) : teste de proximité avec tolérance**
  - **assertNull(object) : vérifie si l'objet est null**
  - **assertNotNull(object) : vérifie si l'objet n'est pas null**
  - **assertSame(expected, actual) : vérifie si les variables référencent le même objet**
  - **assertNotSame(expected, actual) : vérifie que les variables ne référencent pas le même objet**
  - **assertTrue(boolean condition) : vérifie que la condition booléenne est vraie**
  - **……**

Junit : Context de test

- Plusieurs tests d'une même classe initialisent des objets de la même façon
- Création de deux méthodes :
  - Initialisation des objets
  - Libération des ressources
- Méthodes exécutées automatiquement avant et après
- Objets stockés comme attributs

Junit : Context de test

- **Junit3.8 :**
  - **void setUp()**
  - **void tearDown()**
- **Junit4 :**
  - **Méthodes avec annotations @Before ou @After :**
    - Import org.junit.Before;
    - publiques (et plus protected)
    - exécutées avant/aprées chaque méthode de test
    - possibilité d'annoter plusieurs méthodes (ordre d'exécution indéterminée)
  - ✓ **Méthodes avec annotations @BeforeClass et @AfterClass :**
    - publiques et statiques
    - exécutée avant (resp. après) la première (resp. dernière) méthode de test
    - une seule méthode pour chaque annotation

Exemple

```
import junit.framework.* ;

public class JaugeNaturelTest extends TestCase {

    public void testVert() throws Exception {
JaugeNaturel maJauge = new JaugeNaturel(100,200,300);
assertTrue(«le test jauge vert echoue », maJauge.estVert());
    }
}
```

```
import org.junit.*;
import static org.junit.Assert.*;

public class JaugeNaturel2{
    @Test
    public void estVert() throws Exception {
JaugeNaturel maJauge = new JaugeNaturel(100,200,300);
assertTrue(«le test jauge vert echoue », maJauge.estVert());
    }
}
```

Exemple

```
@Before
public void initialiser() throws Exception {
 jauge = new JaugeNaturel(10,20,30);
}
@After
public void nettoyer() throws Exception {
jauge = null;
}
```

Test de programme

- **Junit3.8 :**
  - **Création d'une classe qui regroupe tous les tests (AllTests)**
  - **Implantation d'une méthod public static Test suite() :**
    - Instanciation d'un object de classe TestSuite
    - Ajout des classes de test à exécuter avec la méthode : addTestSuite(Class testClass)

```
import junit.framework.*;
      public class AllTests {
      public static Test suite() {
            TestSuite suite = new TestSuite("Tests");
            suite.addTestSuite(JaugeNaturel1Test.class);
            suite.addTestSuite(JaugeNaturel2Test.class);
      return suite ;
}
   }
```

Test de programme

- **Junit 4 :**
  - **Cration d'une classe qui regroupe tous les tests (AllTests)**
  - **Annotations**

```
import org.junit.runner.RunWith;
import org.junit.runners.Suite;
import org.junit.runners.Suite.SuiteClasses;
@RunWith(Suite.class)
@SuiteClasses(value={
    JaugeNaturel1Test.class,
    JaugeNaturel2Test.class})
public class AllTests {
}
```

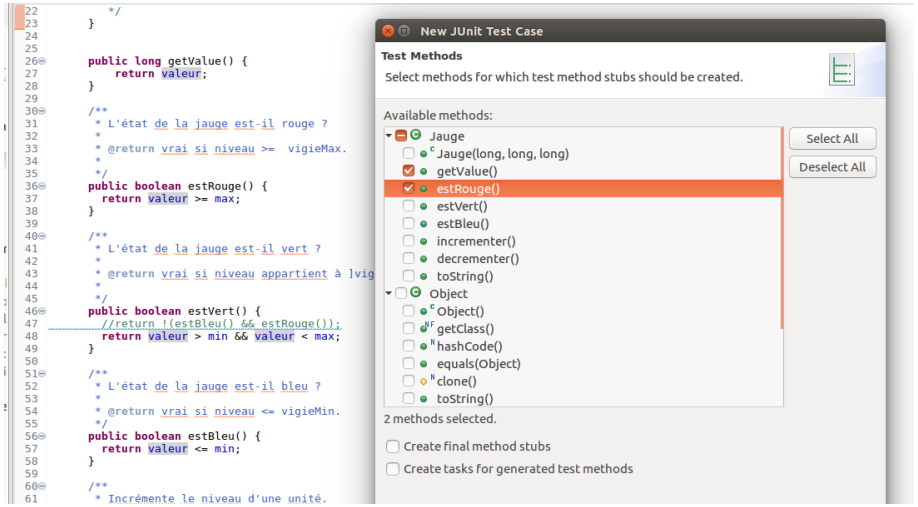
Exécution des tests

- **Junit3.8 :**
  - java -cp ./bin :/usr/share/java/junit-3.8.2.jar
  - junit.textui.TestRunner exemple.AllTests

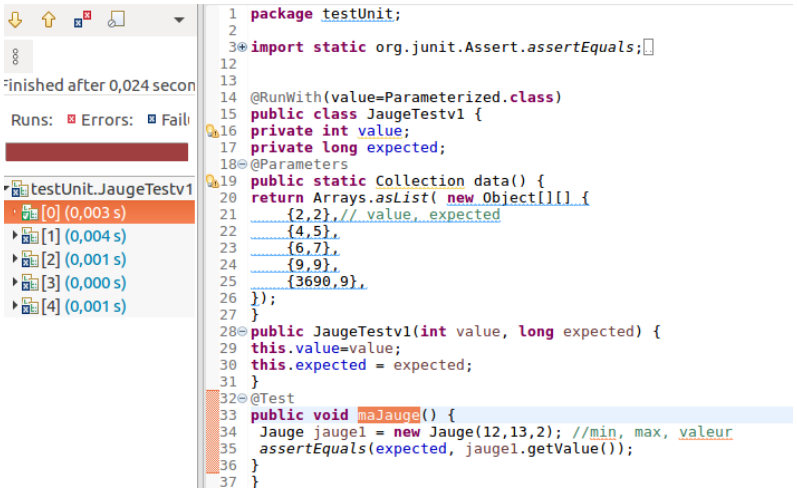
- **Junit4 :**
  - java -cp ./bin :/usr/share/java/junit4-4.3.1.jar
- **Junit5:**
  - API d'excécution jupiter-platform (JUnit 5)
  - https://junit.org/junit5/docs/current/user-guide/#launcher-api

Démo – Junit dans Eclipse

- **Eclipse fournit un assistant pour créer (générer) les tests unitaire**



- **Eclipse intègre un TestRunner pour Junit :**



Autres fonctionnalités

- **Ignorer des tests**

```
@Ignore (“not ready yet / test inutile”) // @Disabled pour Junit5
@Test
public void testIgnore() {
Assert.assertFalse(true);
    }
```

- **Tester une exécution avec une limite dans le temps:**

```
@Test
void timeoutNotExceeded() {
assertTimeout(ofMinutes(), ….
}
```

Autres fonctionnalités

- **Tester la levée d'une exception :**

```
@Test (expected = NullPointerException.class)

public void shouldThrowAnRuntimeException(){
      throw new NullPointerException( “test à interrompre pour lever      une
runtime exception"); }

@Rule
public ExpectedException exceptionRule = ExpectedException.none();

@Test public void whenExceptionThrown__thenRuleIsApplied() {
exceptionRule.expect(ArithmeticException.class);
exceptionRule.expectMessage(“devision par zero”);
calculator.divide(1, 0));

exceptionRule.expect(RuntimeException.class);
exceptionRule.expectMessage(“catch me”);
throw new RuntimeException(“catch me”);
```

Autres fonctionnalités

- **Tester paramétré :**

```
@RunWith(value=Parameterized.class)
public class JaugeTestv1 {
private int value;
private long expected;
@Parameters
public static Collection data() {
   return Arrays.asList( new Object[][] {
        {2,2},// value, expected
        {4,5},
        {6,7},
        {9,9},
        {3690,9},
     });
}
public JaugeTestv1(int value, long expected) {
        this.value=value;
        this.expected = expected;
}
@Test
public void maJauge() {
        Jauge jauge1 = new Jauge(1,13,2); //min, max, valeur
        assertEquals(expected, jauge1.getValue());
}
}
```

Autres fonctionnalités

- **Tester paramétré :**

@RunWith(value=Parameterized.class)

• Exécute tous les tests de la classe avec les données fournies par la méthode annotée par @Parameters

• 5 éléments dans la liste de l'exemple

• Chaque élément est un tableau utilisé comme arguments du constructeur de la classe de test

• Dans l'exemple, les données sont utilisées dans assertEquals

• Equivalent à

```
jauge#0: assertEquals(2, jauge1.getValue() );
jauge#1: assertEquals(4, jauge1.getValue() );
jauge#2: assertEquals(6, jauge1.getValue() );
jauge#3: assertEquals(9, jauge1.getValue() );
jauge#4: assertEquals(3690, jauge1.getValue() );
```

Exécution des tests sous Eclipse - Démo

Méthodes de test : test d'égalité

- void assertEquals(Object expected, Object actual)
- void assertEquals(String message, Object expected, Object actual)
- void assertEquals(long expected, long actual)
- void assertEquals(String message, long expected,long actual)
- void assertEquals(double expected, double actual,double delta)
- void assertEquals(String message, double expected, double actual, double delta)

Méthodes de test : test de condition

- void assertTrue(boolean condition)
- void assertTrue(String message, boolean condition)
- void assertFalse(boolean condition)
- void assertFalse(String message, boolean condition)

Méthodes de test : test objet Null condition

- **void assertNull(Object object)**
- **void assertNull(String message, Object object)**
- **void assertNotNull(Object object)**
- **void assertNotNull(String message, Object object)**

Méthodes de test : test égalité objet

- **void assertSame(Object expected, Object actual)**
- **void assertSame(String message, Object expected, Object actual)**
- **void assertNotSame(Object unexpected, Object actual)**
- **void assertNotSame(String message, Object unexpected, Object actual)**

Méthodes de test : test objet Null condition

- **void assertNull(Object object)**
- **void assertNull(String message, Object object)**
- **void assertNotNull(Object object)**
- **void assertNotNull(String message, Object object)**