Atelier
Tests et intégration continue avec Eclipse

Julien Nauroy

Ingénieur Confirmé CDD Inria

Laboratoire de Recherche en Informatique

Université Paris Sud

23/05/2013

Qu’allons-nous voir ?

1. Création d’un projet simple en Java
2. Rédaction de tests avec Junit 3
3. Couverture de code
4. Automatisation des tests
5. Intégration continue

- Projet : déterminer le type d’un triangle
- Le tout en 2h !

Que n’allons-nous pas voir ?

- Beaucoup de choses !
  - Les bonnes pratiques de programmation Java
  - Junit 4 (Java 1.5+ => Annotations)
  - Les esclaves avec Jenkins
  - La couverture de code dans Jenkins
- Tutoriel en largeur plutôt qu’en profondeur
- Le contenu ne reflète pas toujours les meilleures pratiques

Prérequis techniques

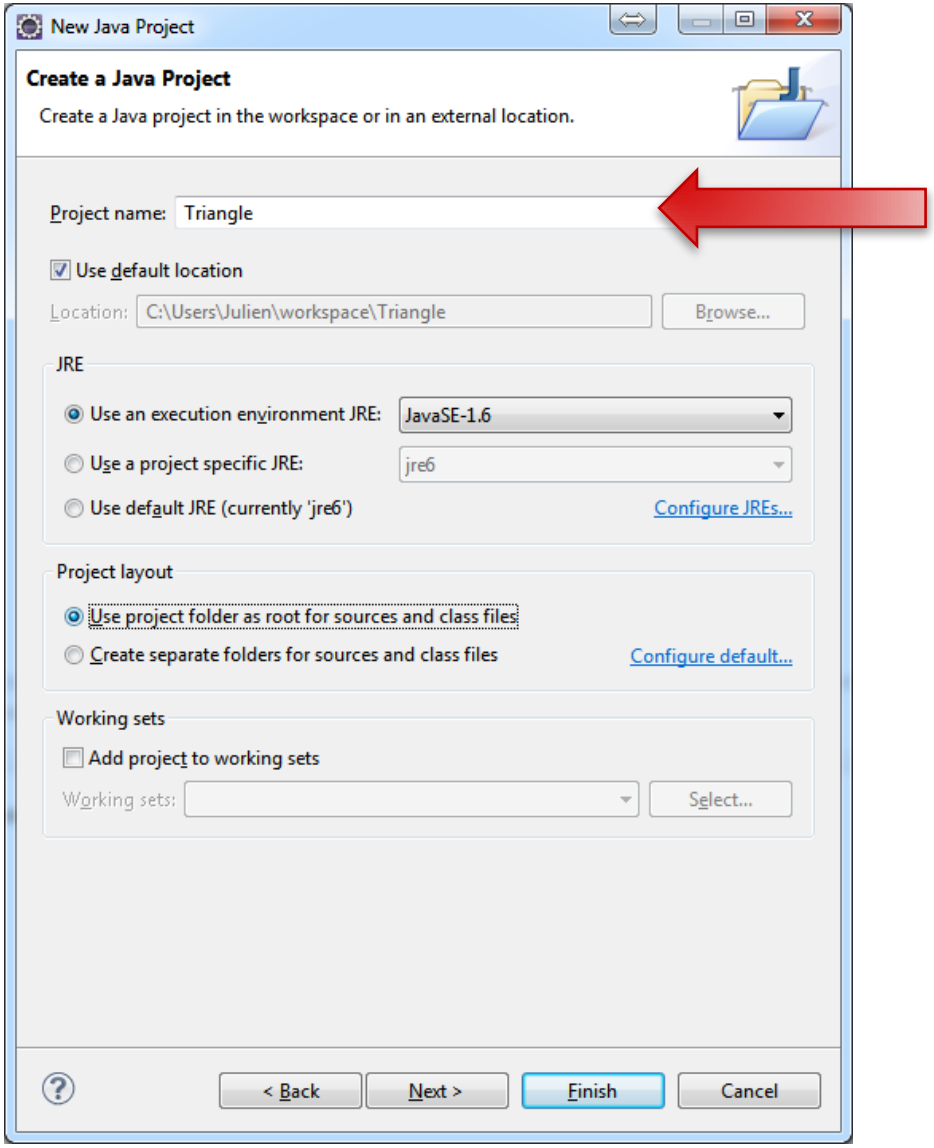
- Installer Java 1.6 – 1.7
- Installer Ant
- Installer Eclipse Juno 4.2
- Télécharger Jenkins (http://jenkins-ci.org/)

Utilisateurs de Windows: mettez à jour les variables système

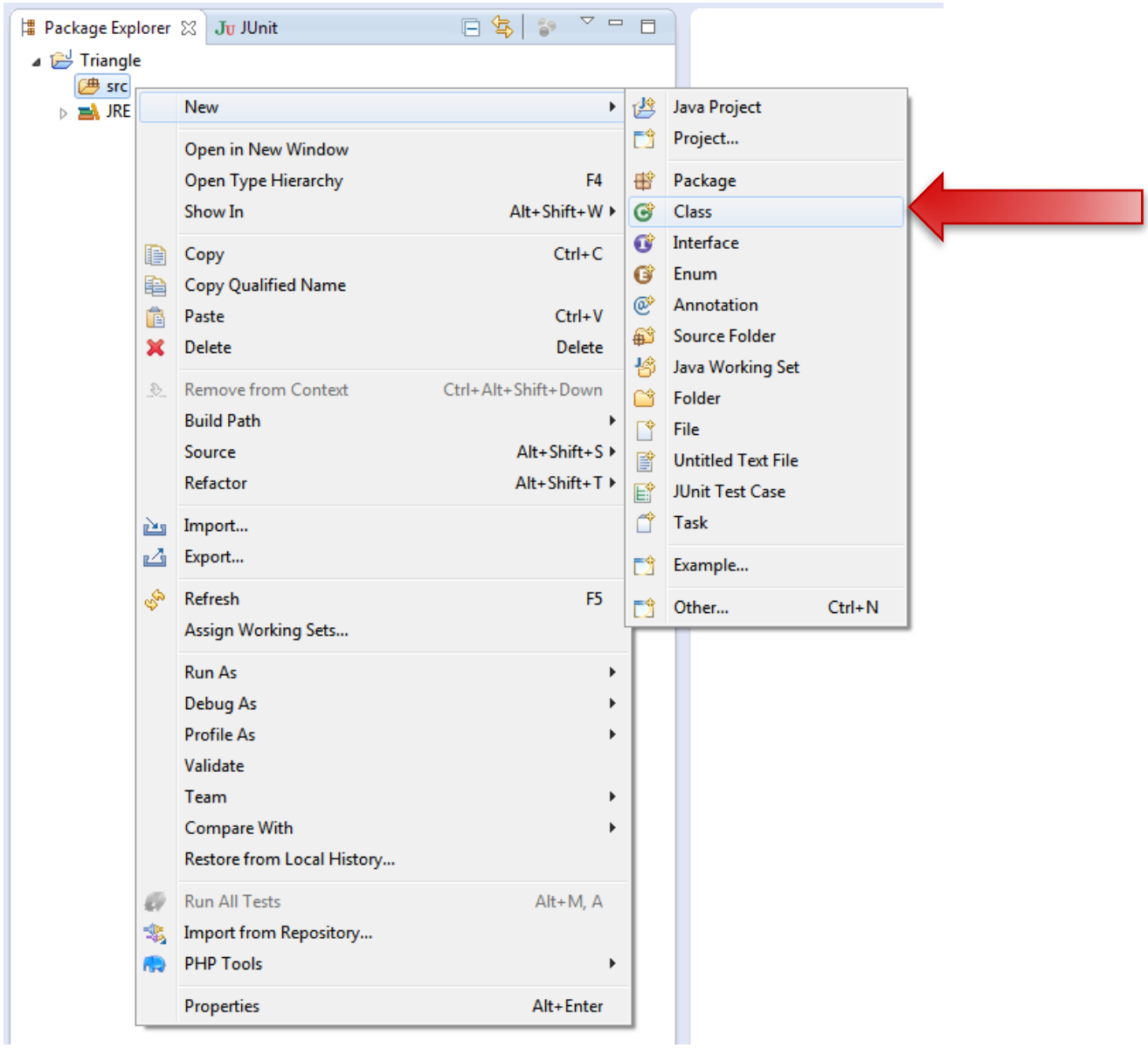
1. Création d’un projet simple

Création d'un projet sous Eclipse

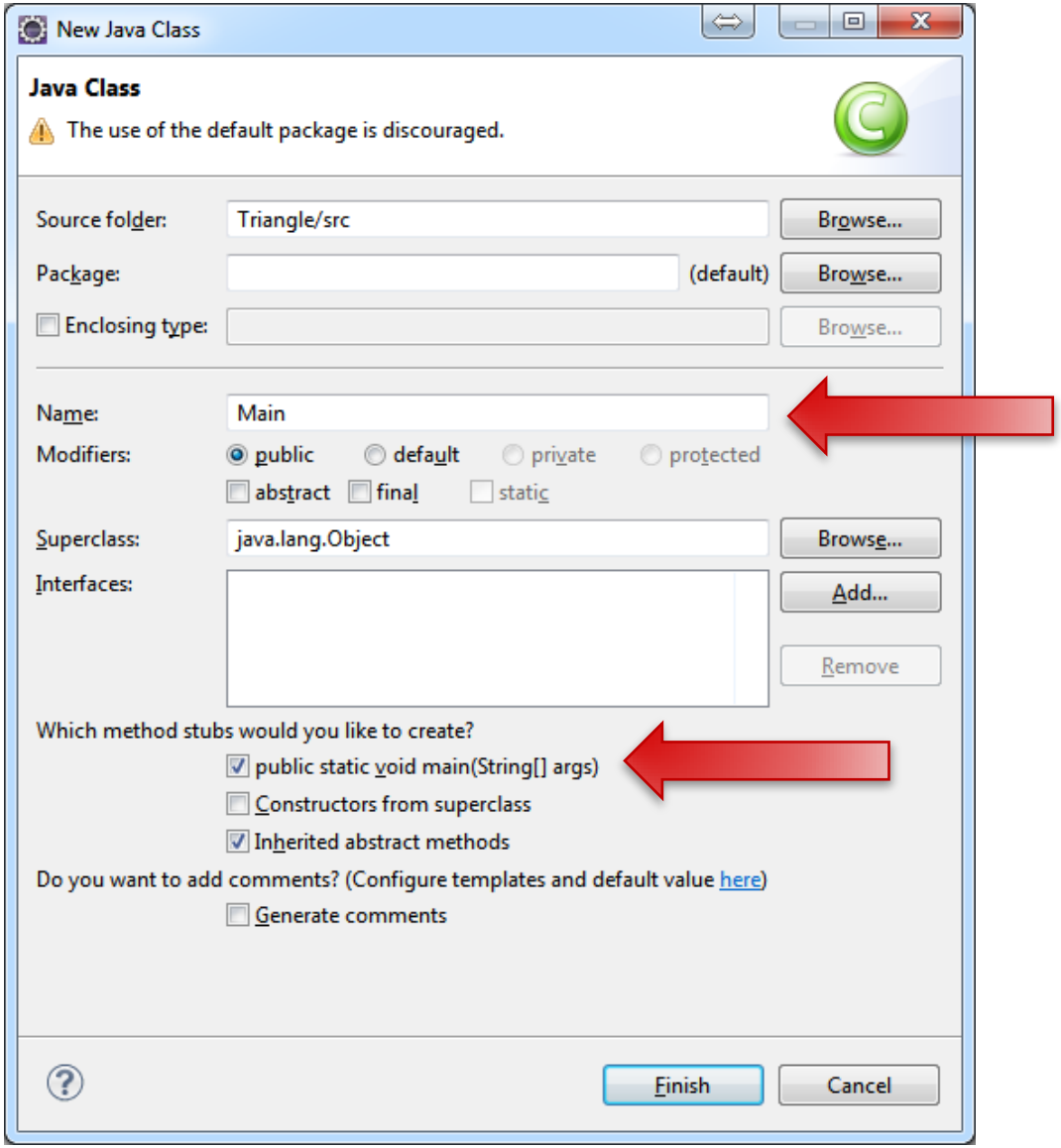
- File -> New -> Project
  - Java -> Java Project



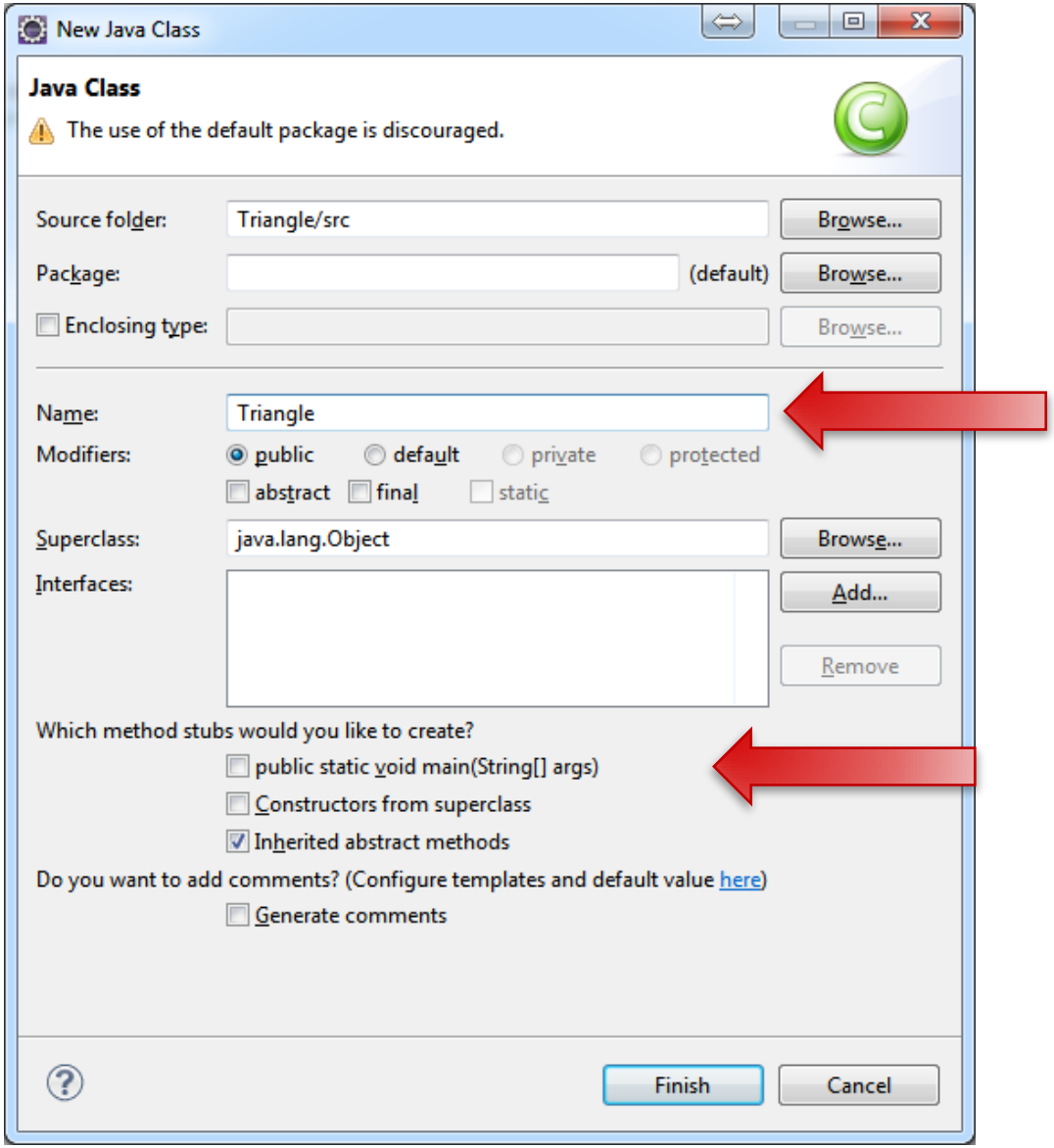
Création d'une classe



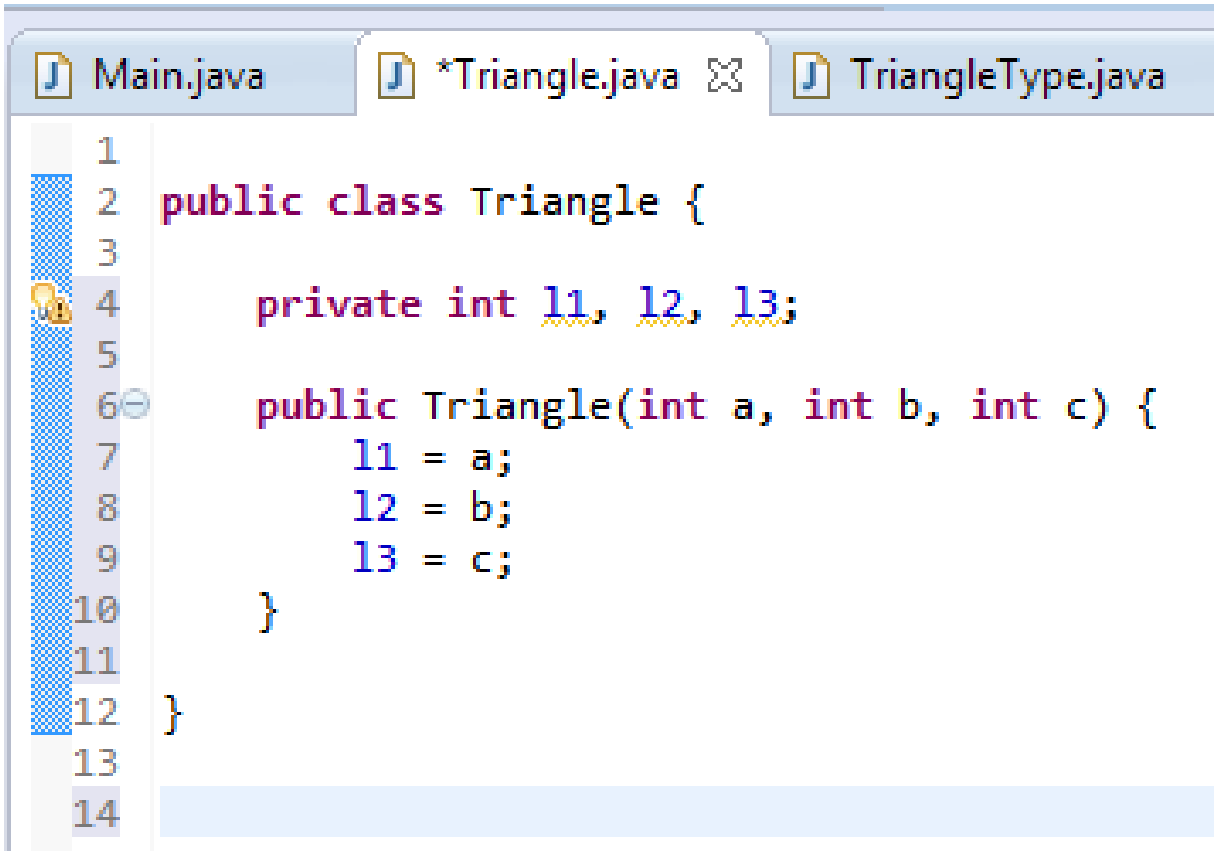
Classe Main


New Java Class
Java Class
The use of the default package is discouraged.
Source folder:
Triangle/src
Browse...
Package:
(default)
Browse...
Enclosing type:
Browse...
Name:
Main
Modifiers:
public
default
private
protected
abstract
final
static
Superclass:
java.lang.Object
Browse...
Interfaces:
Add...
Remove
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
Inherited abstract methods
Do you want to add comments? (Configure templates and default value here)
Generate comments
Finish
Cancel

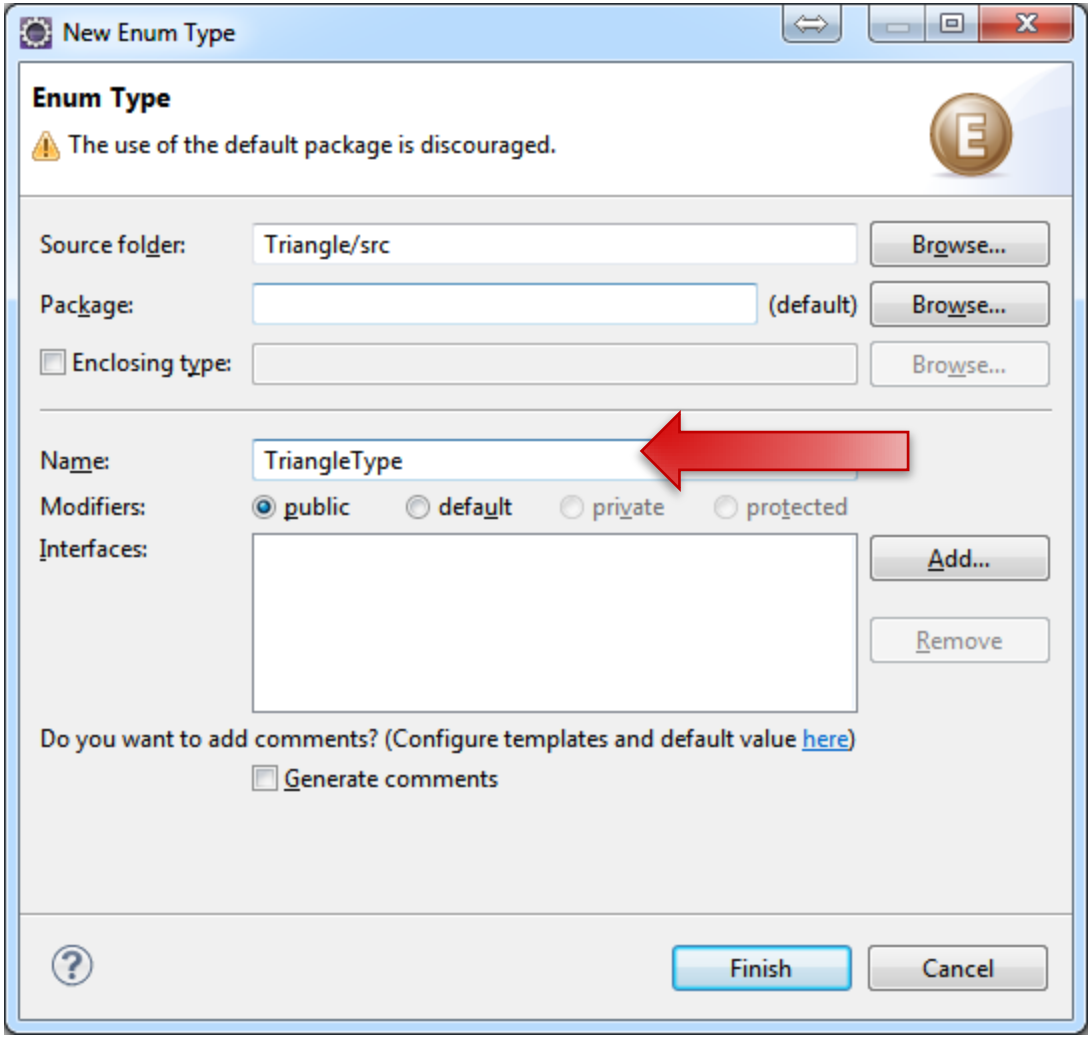
Classe Triangle



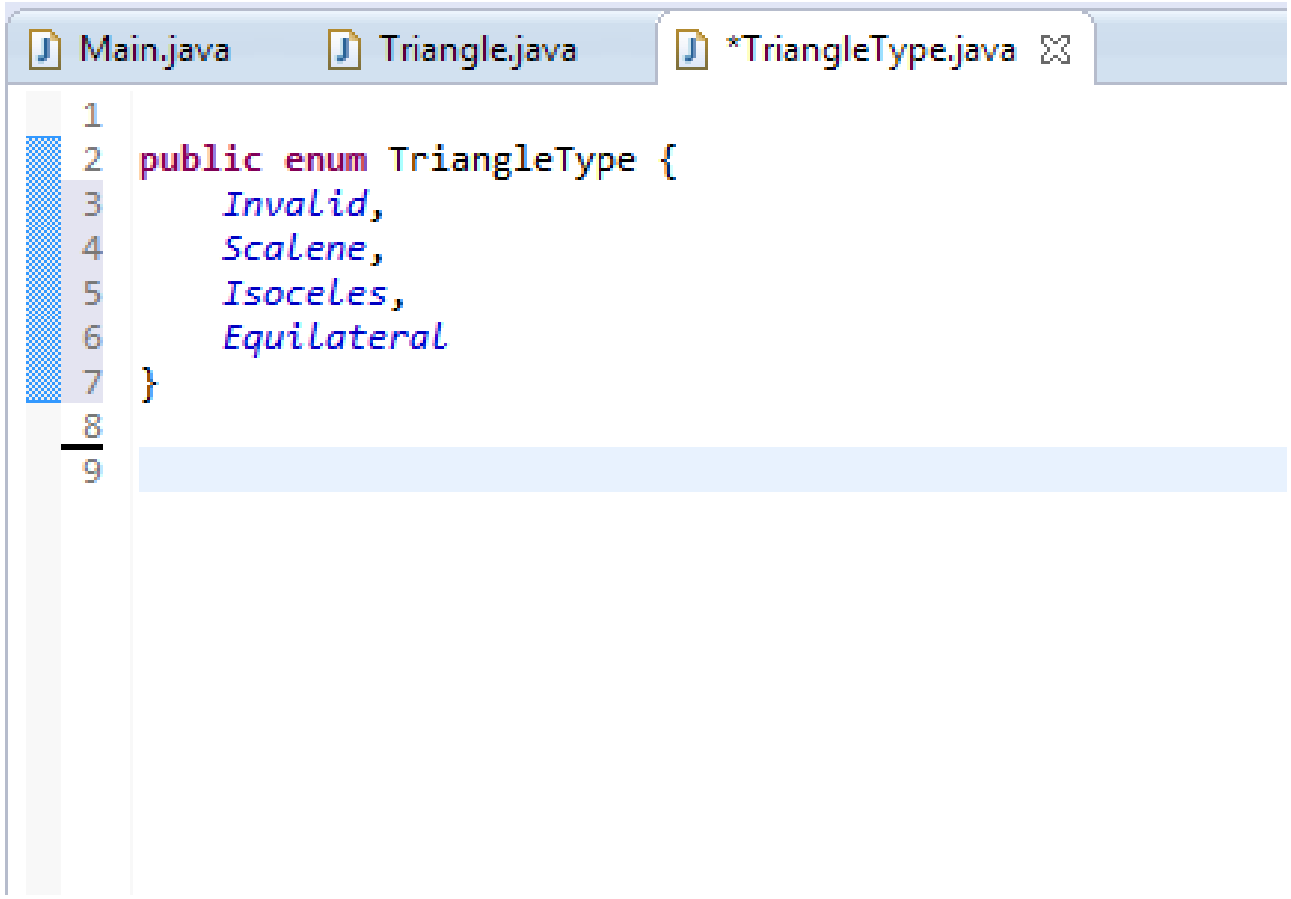
Contenu de Triangle



Enumération TriangleType



Contenu de TriangleType



Exercice

- Créez une méthode getType() qui renvoie le type de triangle
- Quels sont les cas valides ?

Exercice

- Créez une méthode getType() qui renvoie le type de triangle
- Quels sont les cas valides ?
  - Triangle quelconque (scalène)
  - Triangle isocèle
  - Triangle équilatéral
- Quels sont les cas invalides ?

Exercice

- Créez une méthode getType() qui renvoie le type de triangle
- Quels sont les cas valides ?
- Quels sont les cas invalides ?
  - Une longueur est négative ou nulle
  - Une longueur n’est pas numérique (ou entière)
  - Le triangle est plat
  - Une longueur est supérieure à la somme des 2 autres

Exercice

- Créez une méthode getType() qui renvoie le type de triangle

```
public TriangleType getType() {
    return TriangleType.Invalid;
}
```

- Complétez

Tests simples

- Modifiez la fonction main comme ceci :

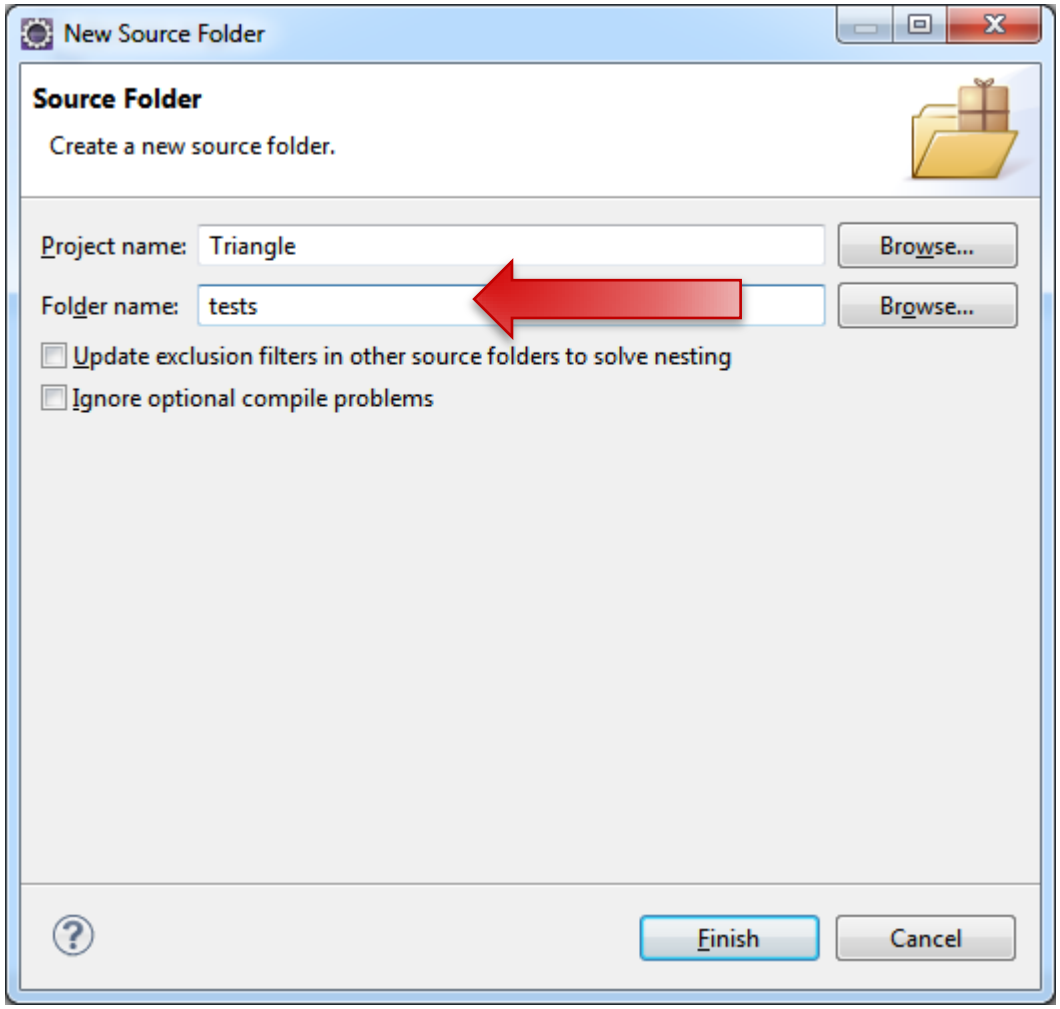
```
public static void main(String[] args) {
    Triangle t1 = new Triangle(3, 4, 5);
    System.out.println("Triangle t1 is " + t1.getType());

    Triangle t2 = new Triangle(0, 0, 0);
    System.out.println("Triangle t2 is " + t2.getType());

    Triangle t3 = new Triangle(2, 3, 7);
    System.out.println("Triangle t3 is " + t3.getType());
}
```

- Exécutez. Le résultat est-il correct ?

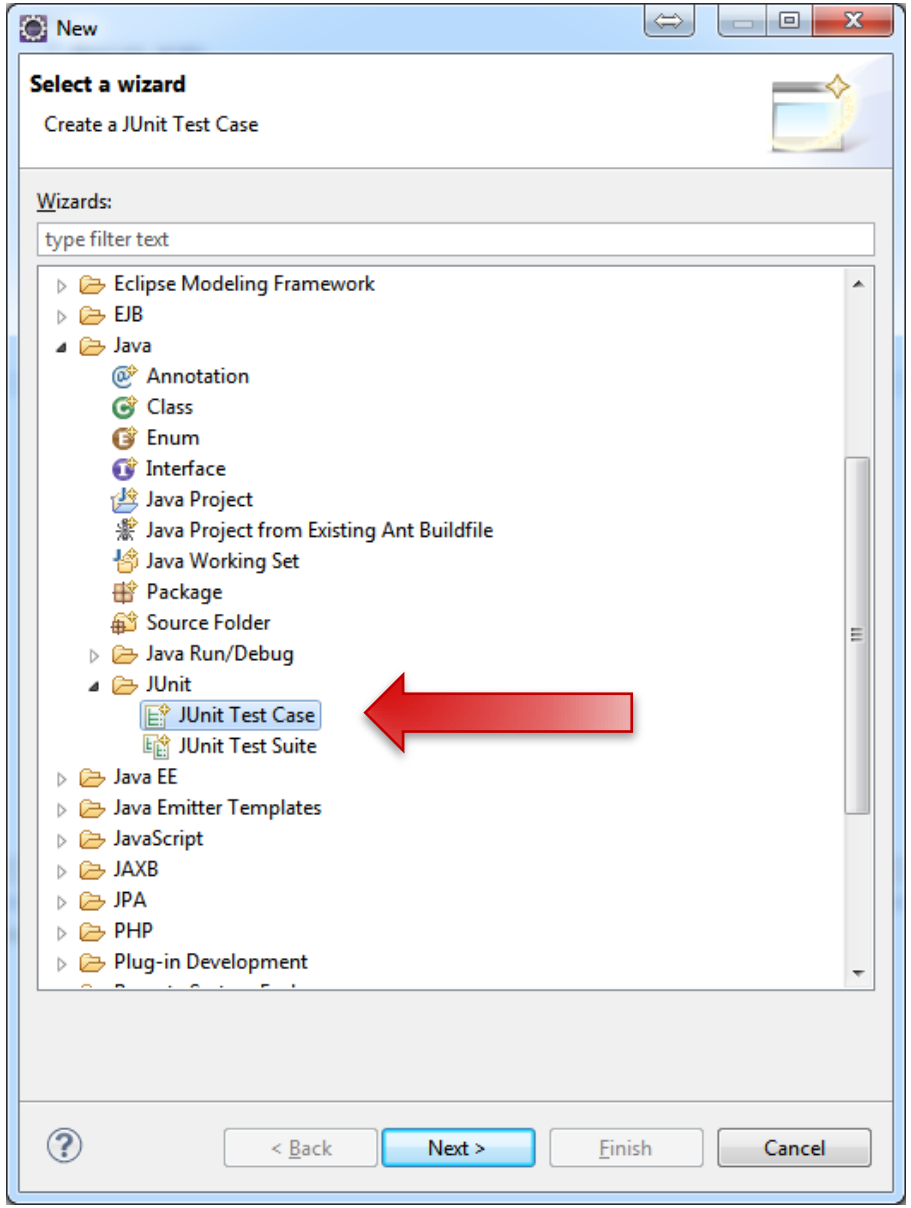
2. Rédaction de tests avec JUnit

Création d'un dossier de tests

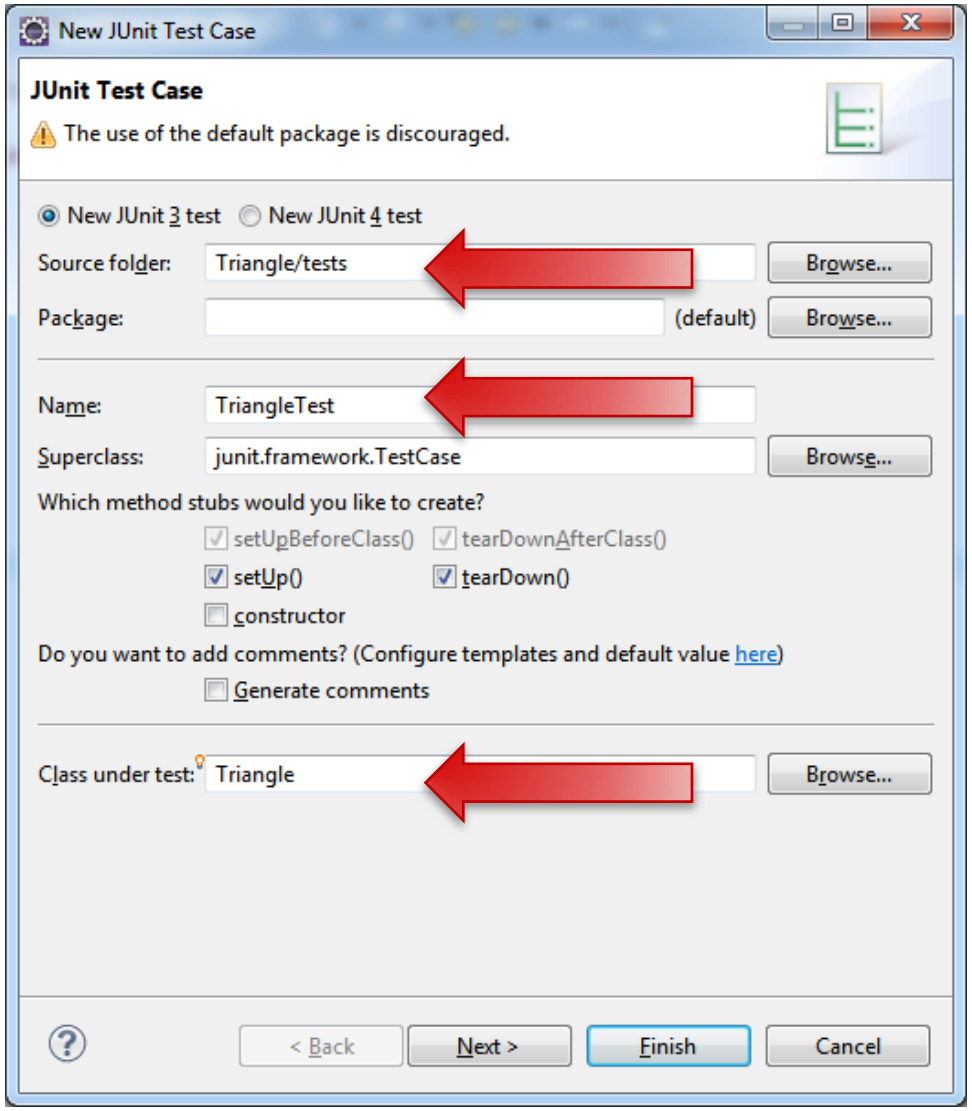
- New -> Source Folder



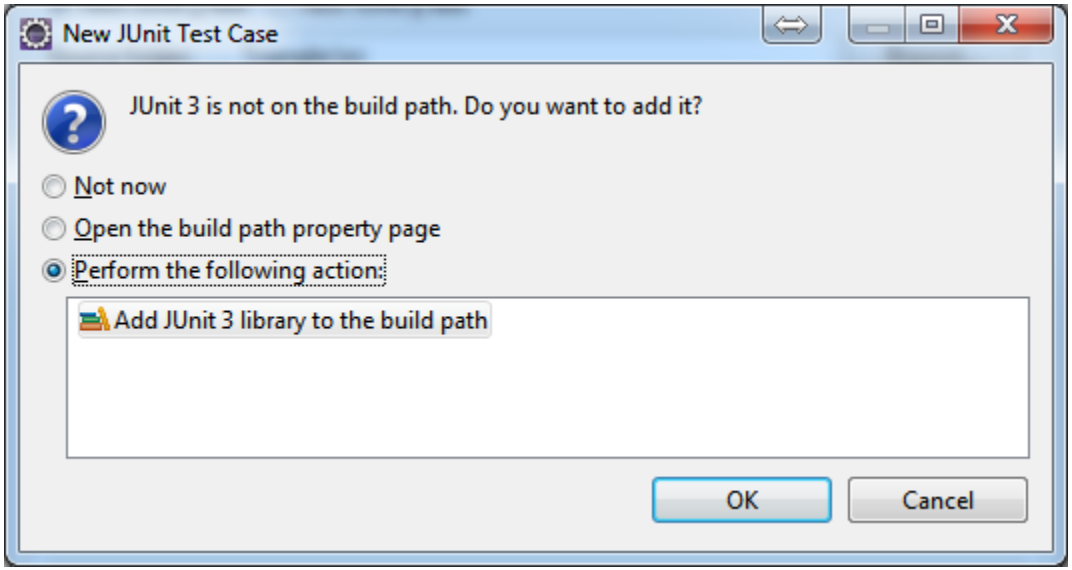
Création d'un test



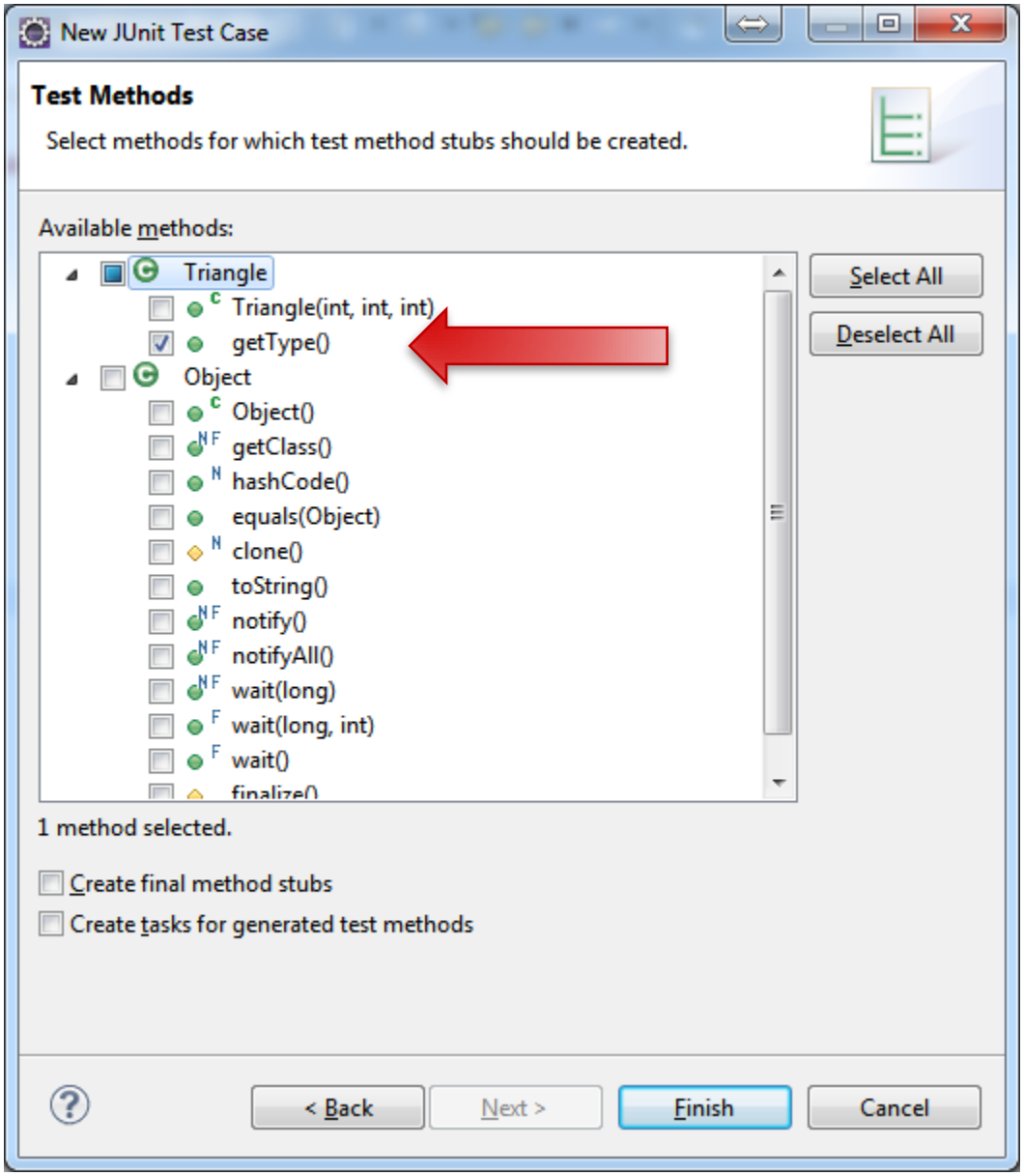
Paramètres du test



Ajout de Junit au build path



Choix des méthodes à tester



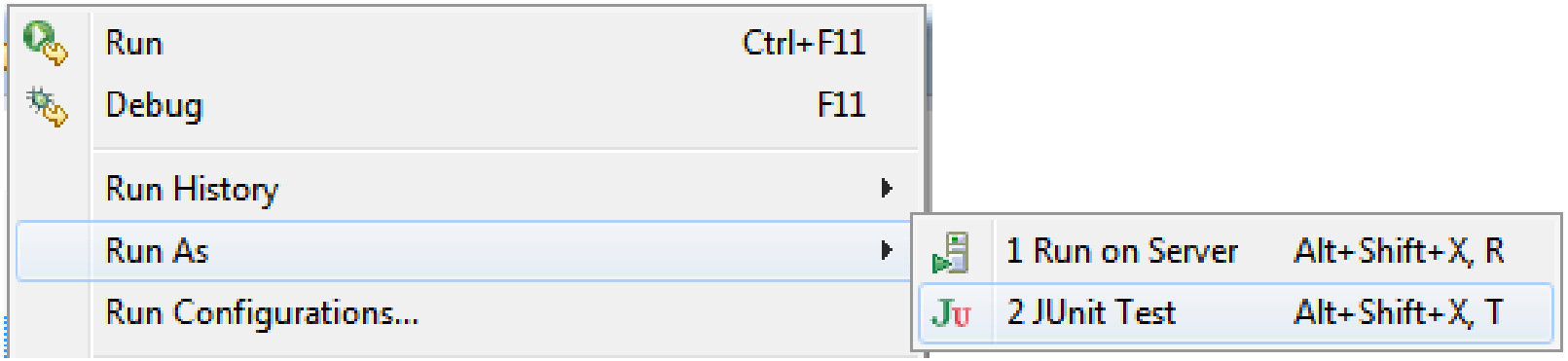
Configuration de départ

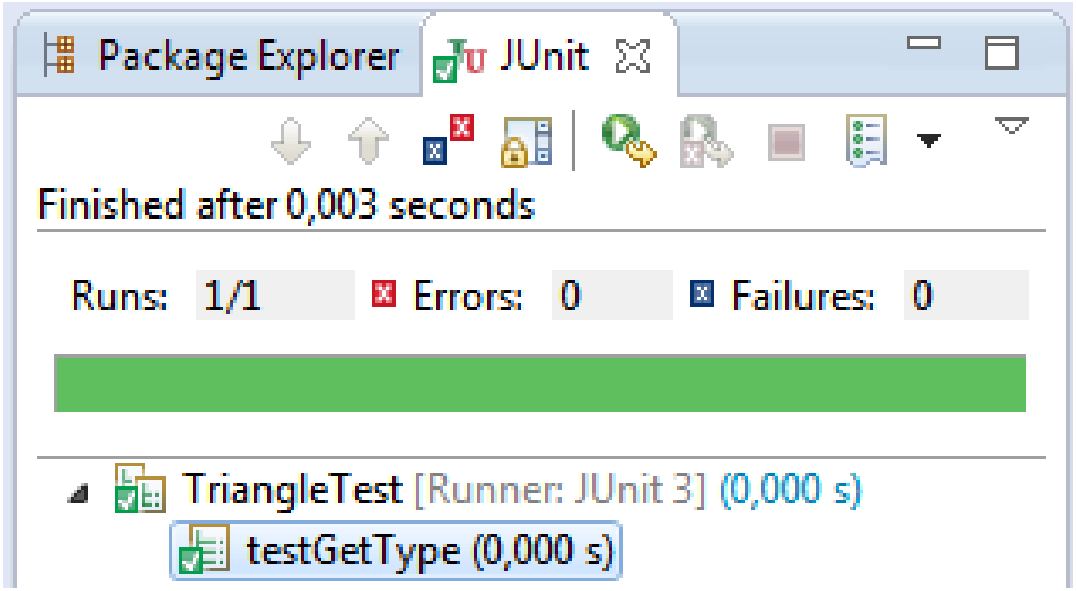
```
import junit.framework.TestCase;


public class TriangleTest extends TestCase {

    protected static void setUpBeforeClass() throws Exception {
    }

    protected static void tearDownAfterClass() throws Exception {
    }

    protected void setUp() throws Exception {
        super.setUp();
    }

    protected void tearDown() throws Exception {
        super.tearDown();
    }

    public void testGetType() {
        fail("Not yet implemented");
    }

}
```

Un premier test

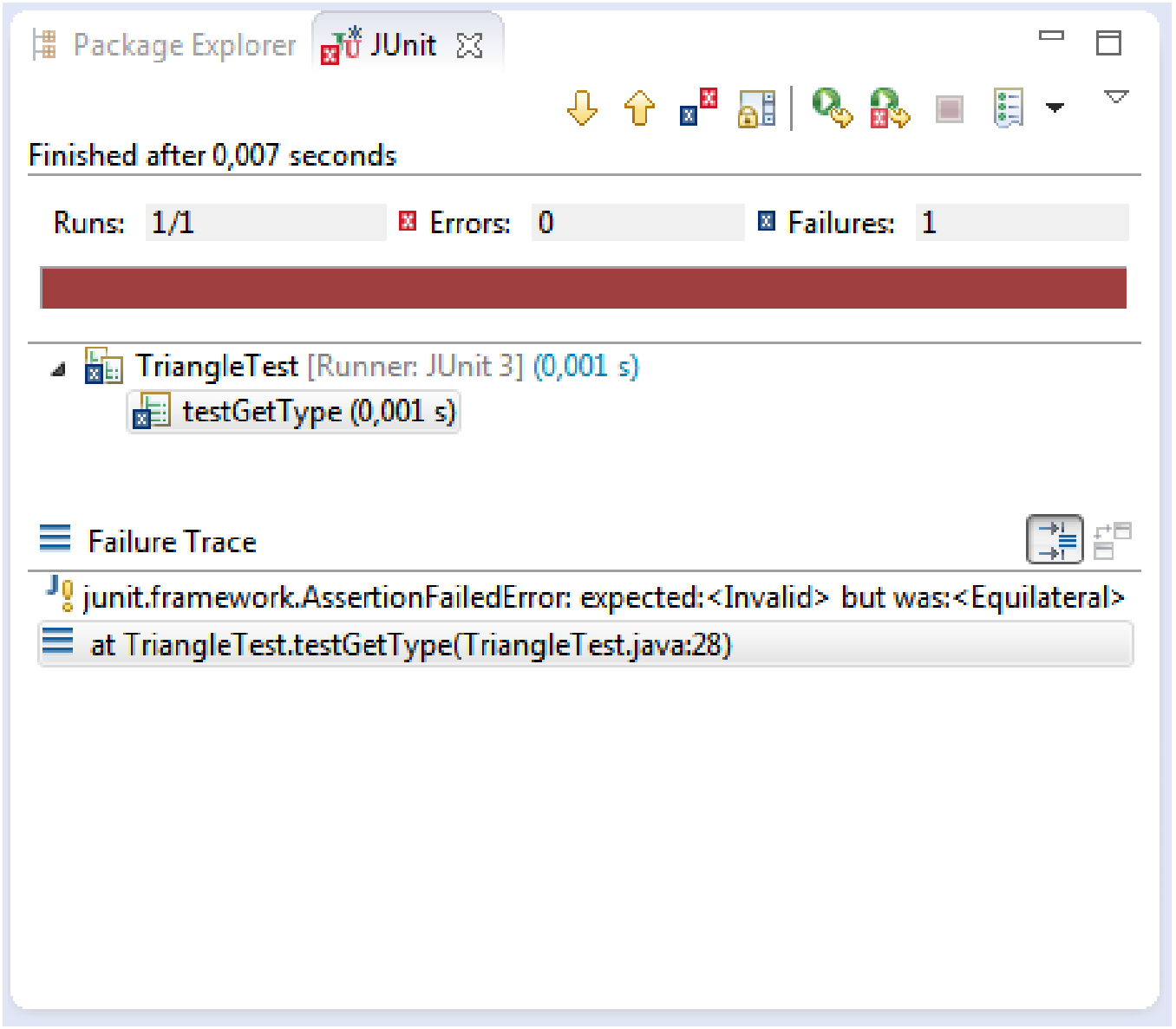
```
public void testGetType() {
    Triangle t1 = new Triangle(3, 4, 5);
    assertEquals(t1.getType(), TriangleType.Scalene);
}
```

Exécuter les tests





Et si on se trompe ?



Exercice

- Créez des tests pour vérifier que votre fonction est valide
- Quelques exemples :
  - (3, 4, 5) => quelconque
  - (5, 7, 5) => isocèle
  - (9, 9, 9) => équilatéral
  - (0, 0, 0) => invalide
  - (1, 2, 3) => invalide
  - (5, 6, 0) => invalide
  - (7, 7, -1) => invalide

Pour aller plus loin…

- Autres assertions :
  - assertTrue, assertFalse
  - assertNull, assertNotNull
  - assertSame, assertNotSame
- TestSuite : ensemble de TestCase
- Fixtures
  - setUp, tearDown
  - setUpBeforeClass, tearDownAfterClass

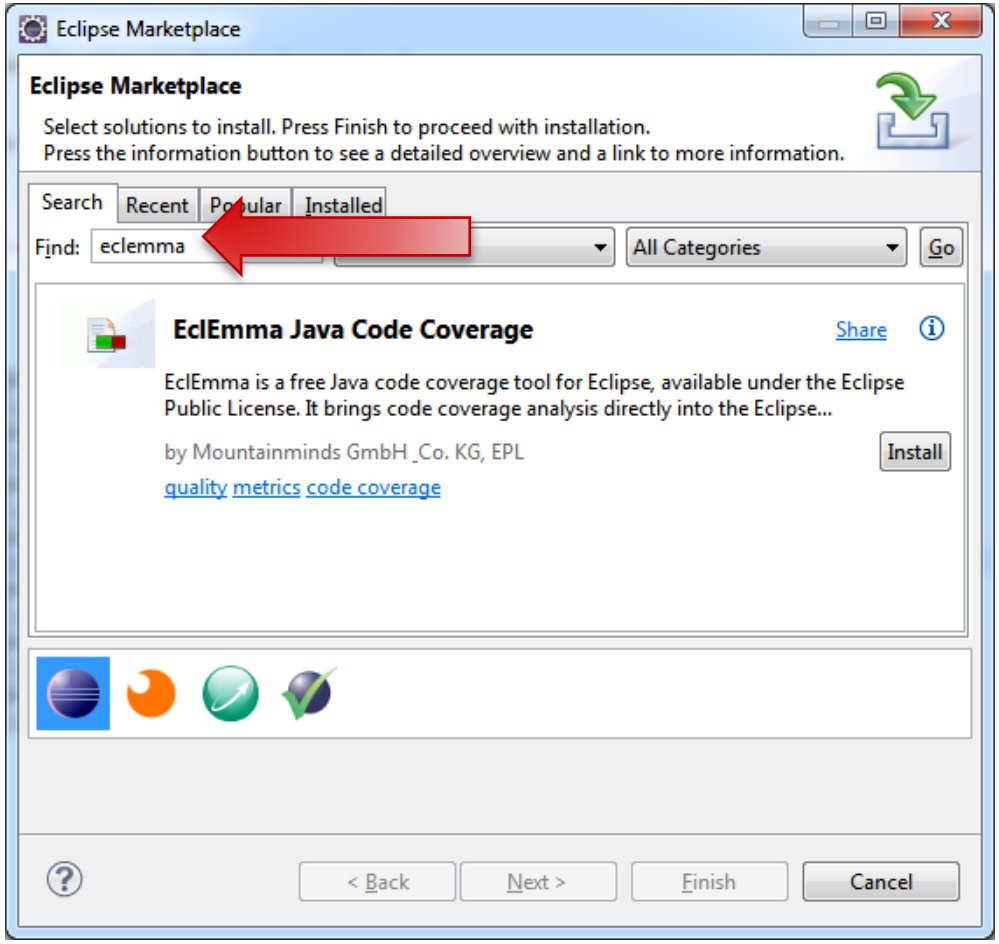
Pour aller plus loin…

- Beaucoup de problématiques spécifiques :
  - Besoin de données externes
    - Base de données
    - Fichier(s)
    - Connexions distantes
  - Effets de bord
    - Modifications de BdD, de fichiers
  - Besoin d'interaction de l'utilisateur
    - Saisie de données
    - Manipulation de l'interface graphique
  - Liens entre fonctions : testCreate avant testDelete

3. Couverture de code

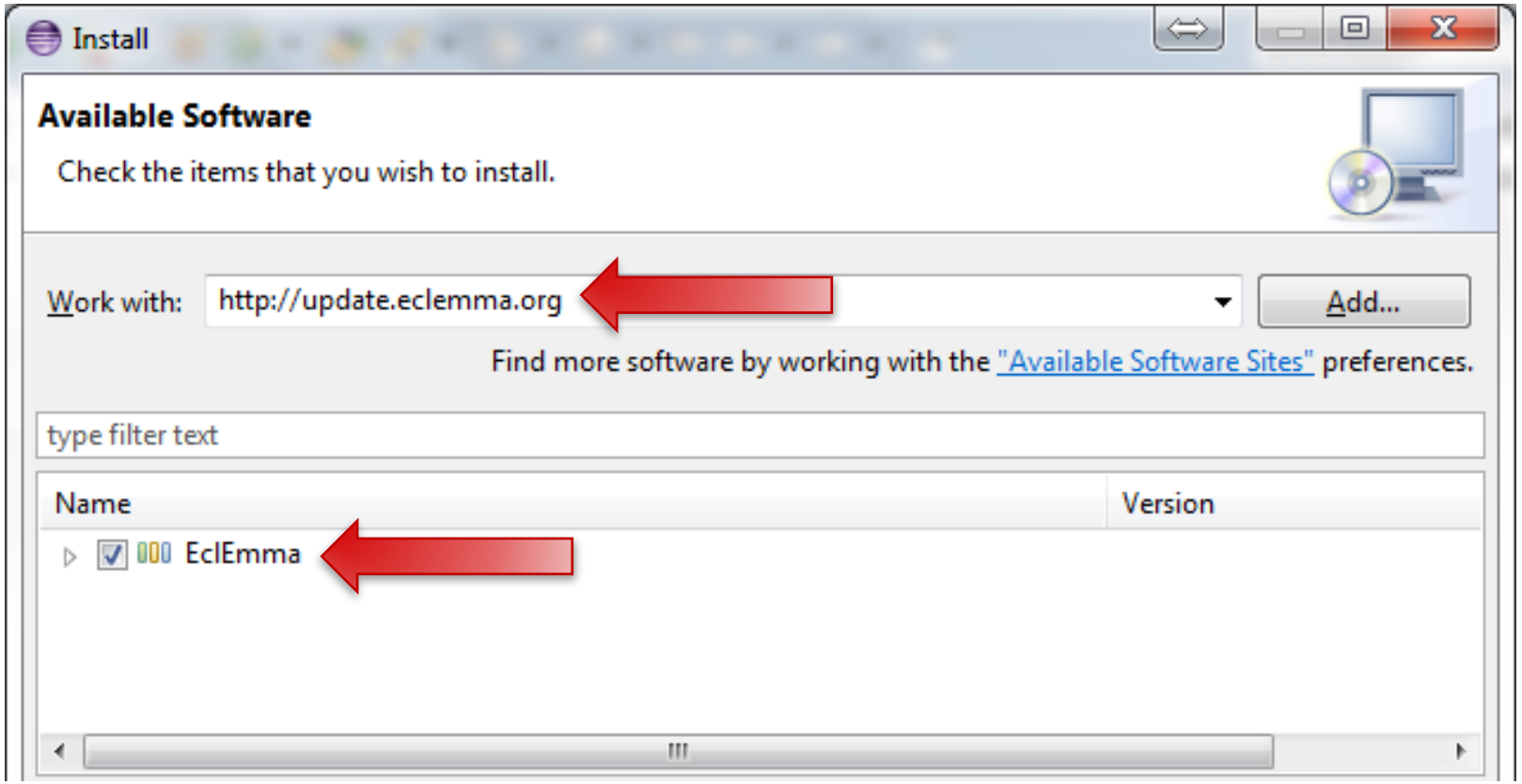
Installation d'EclEMMA

- Help -> Eclipse Marketplace

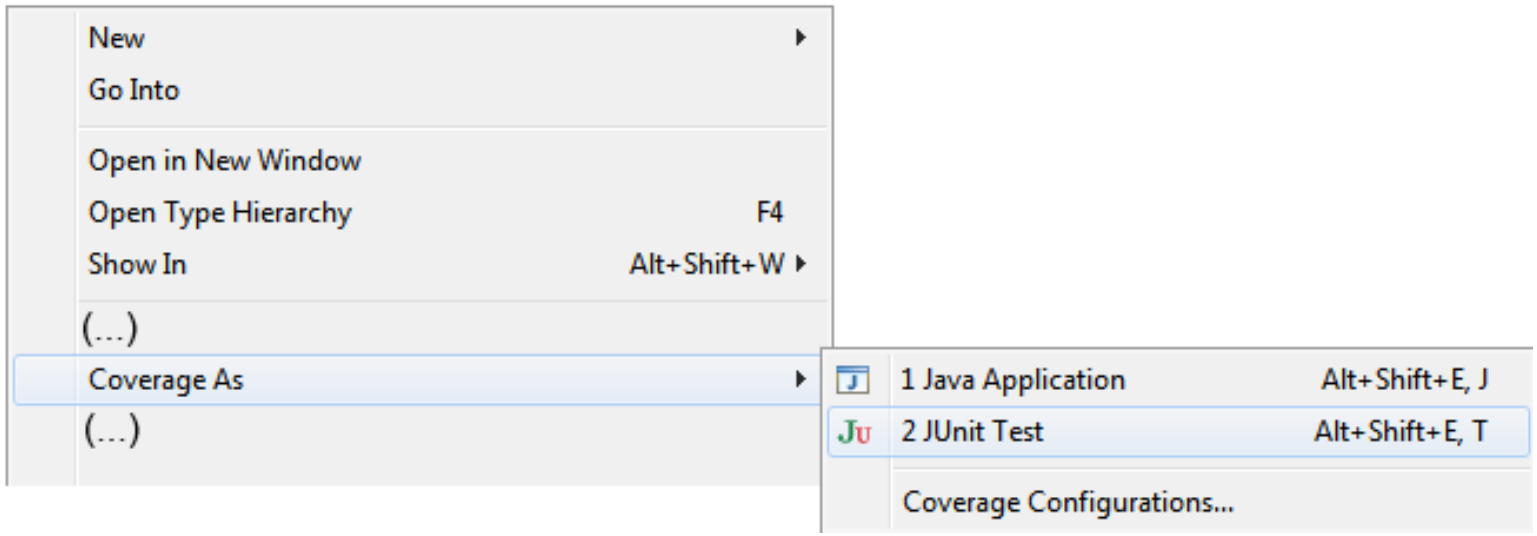


Installation d'EclEmma (2)

- Si le Marketplace n'est pas disponible :
- Help -> Install new Software



Utilisation d'EclEMMA



Résultat

```java
public Triangle(int a, int b, int c) {
    l1 = a;
    l2 = b;
    l3 = c;
}

public TriangleType getType() {
    // Check if the lengths are strictly positive
    if(l1 <= 0 || l2 <= 0 || l3 <= 0) {
        return TriangleType.Invalid;
    }

    // Check if one length is equal to or greater than
    // the sum of the two others
    if(l1 >= l2 + l3 || l2 >= l1 + l3 || l3 >= l2 + l1) {
        return TriangleType.Invalid;
    }

    // Determine if the triangle is equilateral,
    // isoceles or scanele depending on the number
    // of equal lengths
    int equalLengths = 0;
    if (l1 == l2) {
        equalLengths++;
    }
    if (l1 == l3) {
        equalLengths++;
    }
    if (l3 == l2) {
        equalLengths++;
    }

    if(equalLengths == 3) {
        return TriangleType.Equilateral;
    } else if(equalLengths == 1) {
        return TriangleType.Isoceles;
    }
    return TriangleType.Scalene;
}
```

```java
public void testGetType() {
    Triangle t1 = new Triangle(3, 4, 5);
    assertEquals(t1.getType(), TriangleType.Scalene);
    Triangle t2 = new Triangle(5, 7, 5);
    assertEquals(t2.getType(), TriangleType.Isoceles);
    Triangle t3 = new Triangle(9, 9, 9);
    assertEquals(t3.getType(), TriangleType.Equilateral);
    Triangle t4 = new Triangle(0, 0, 0);
    assertEquals(t4.getType(), TriangleType.Invalid);
    Triangle t5 = new Triangle(1, 2, 3);
    assertEquals(t5.getType(), TriangleType.Invalid);
    Triangle t6 = new Triangle(5, 6, 0);
    assertEquals(t6.getType(), TriangleType.Invalid);
    Triangle t7 = new Triangle(7, 7, -1);
    assertEquals(t7.getType(), TriangleType.Invalid);
}
```

Exercice

- Essayez d’atteindre 100% de couverture de code pour la fonction getType()

Pour aller plus loin

- Test paramétrique
  - (0, 0, 0) => Invalide
  - (4, 5, 6) => Scalene
  - Etc
  - Junit 4 seulement ?
- Ordonnancement des tests
  - Pas avec JUnit ?
  - Pratique pour tester suppression après création
    - Mais pas recommandé

4. Automatisation

Rendre les tests indépendants

- Les tests fonctionnent bien avec Eclipse…
- … Mais il faut pouvoir s'en passer !
- Plusieurs façons : ant, maven, …
- Ici, nous allons utiliser ant

Ant

« ***Ant*** est un logiciel créé par la fondation Apache qui vise à automatiser les opérations répétitives du développement de logiciel telles que la compilation, la génération de documents (Javadoc) ou l'archivage au format JAR, à l'instar des logiciels Make. »

Wikipedia

Installer Ant

- Télécharger à l’adresse suivante:
  http://ant.apache.org/bindownload.cgi
- Le décompresser et le référencer dans le path

  OU

  - apt-get install ant
  - Ou équivalent

Exécution manuelle des tests

- Création d'un répertoire « lib »
- Récupération du .jar de Junit
  - eclipse/plugins/org.junit_3.xxx/junit.jar
- Exécution manuelle du test

  > mkdir build

  > javac -d build/ -cp lib/junit.jar src/*.java tests/*.java

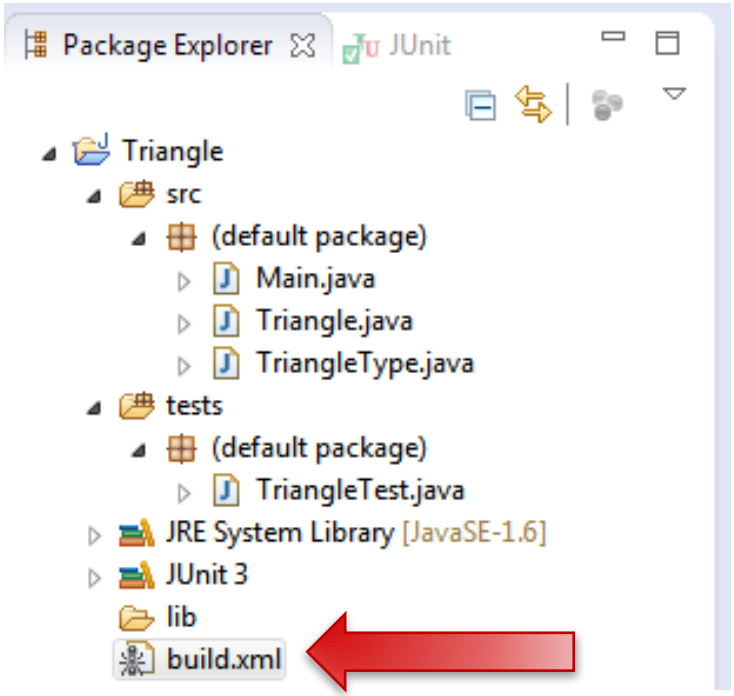
  > java -cp build/;lib/junit.jar junit.textui.TestRunner TriangleTest

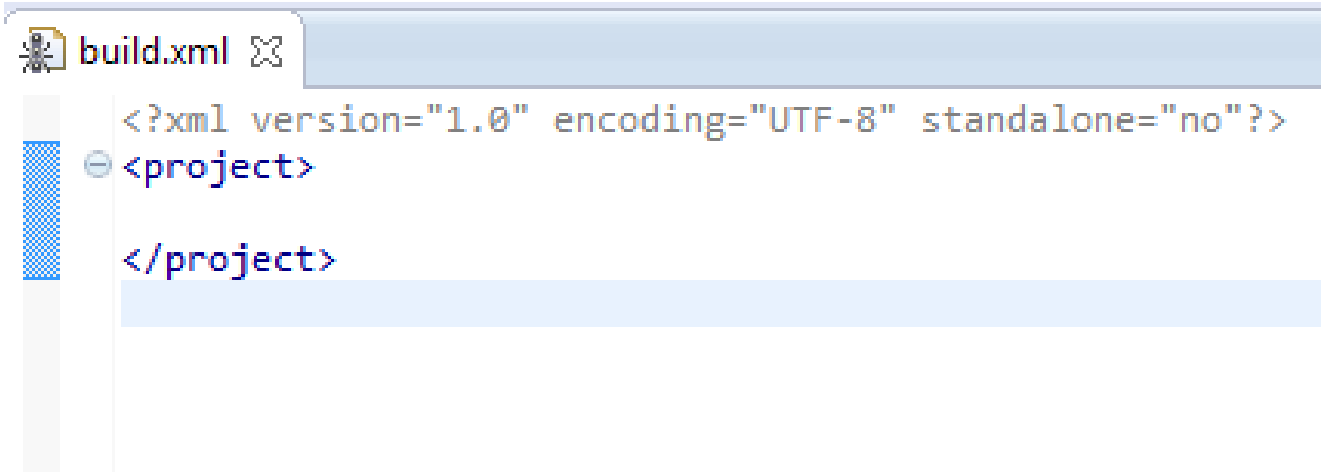
  .

  Time: 0,001

  OK (1 test)

Création d'un fichier build.xml





Création de cibles

build.xml

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<project>
    <target name="clean">
        <delete dir="build"/>
    </target>

    <target name="compile">
        <mkdir dir="build"/>
        <javac destdir="build" srcdir="src;tests" classpath="lib/junit.jar" />
    </target>

    <target name="tests" depends="compile">
        <junit>
            <classpath>
                <pathelement location="lib/junit.jar"/>
                <pathelement location="build/"/>
            </classpath>
            <formatter type="xml"/>
            <test name="TriangleTest" outfile="result"/>
        </junit>
    </target>
</project>
```

Essai

```
> ant compile
Buildfile: (...)/build.xml

compile:
    [mkdir] Created dir: (...)/build
    [javac] (...)/build.xml:9: warning: (...)
    [javac] Compiling 4 source files to (...)/build

BUILD SUCCESSFUL
Total time: 0 seconds
```

Essai (2)

> ant tests

(...)

tests:

BUILD SUCCESSFUL

Total time: 0 seconds

- Nouveau fichier : result.xml

Pour aller plus loin…

- Code coverage :
  - http://stackoverflow.com/questions/52984/how-do-i-generate-emma-code-coverage-reports-using-ant
  - http://www.ibm.com/developerworks/java/library/j-cobertura/

5. Intégration continue

Mise en place du dépôt SVN

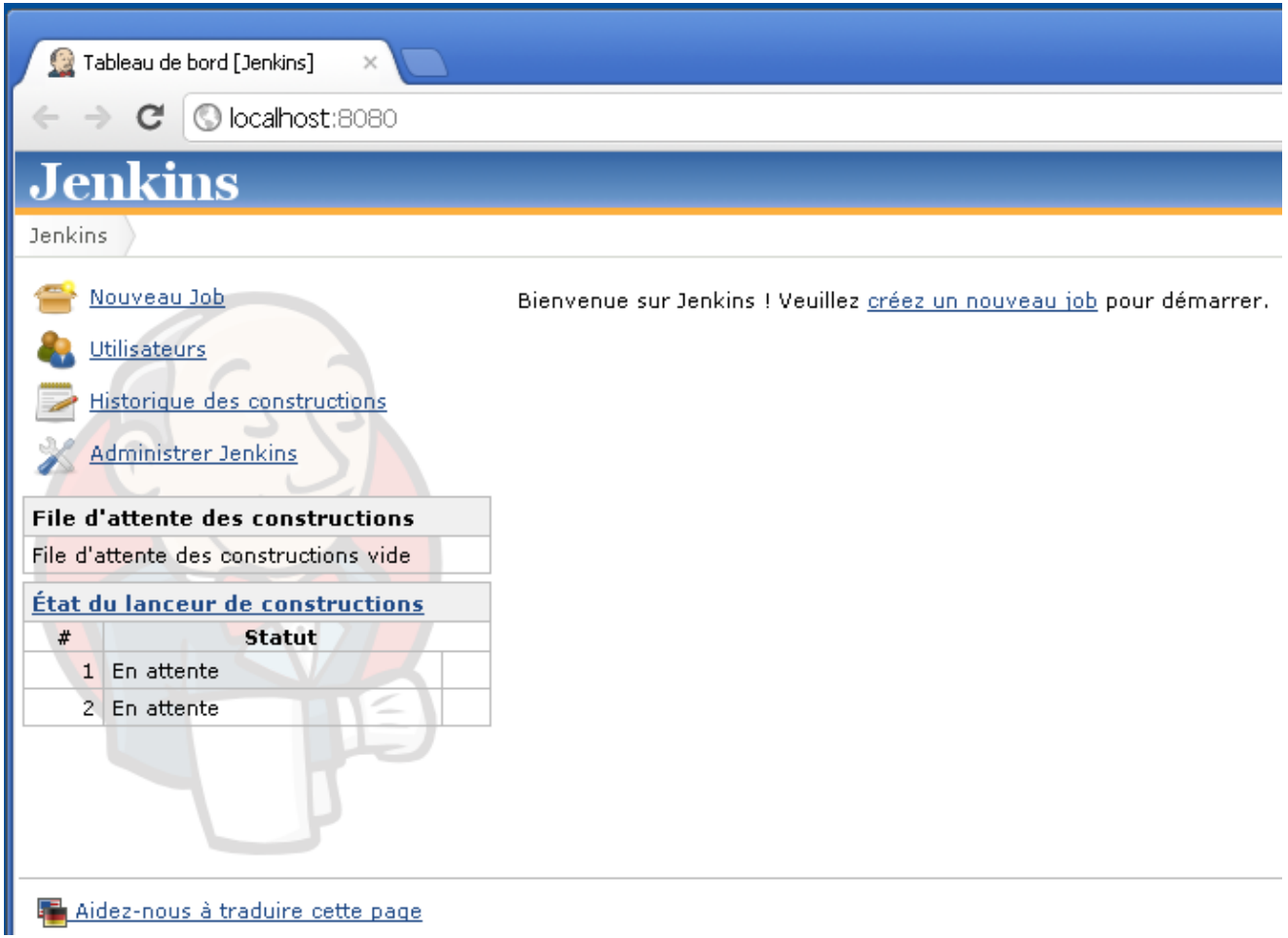
- Utilisation d'un dépôt existant
  - https://www.lri.fr/svn/dev/atelierloops/pX
    - Remplacer X par un numéro entre 1 et 30
    - Login atelier/ password loops, etc
  - Ou le vôtre (CSV, SVN, Git…)

Exercice :

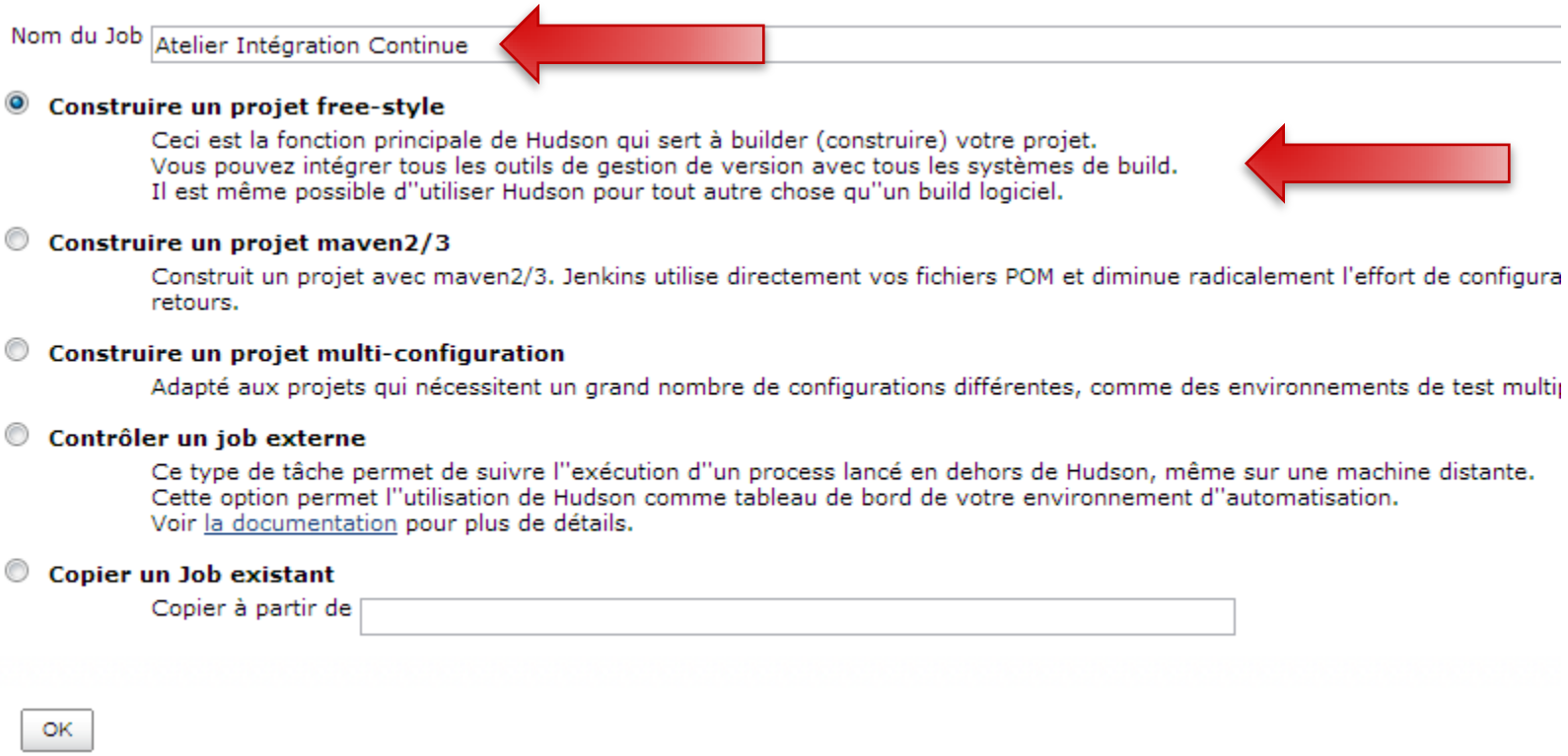
- Versionnez votre projet dans un dépôt

Mise en place de Jenkins

- Télécharger Jenkins depuis jenkins-ci.org
  - Version « binaire », pas « WAR »
- L'installer

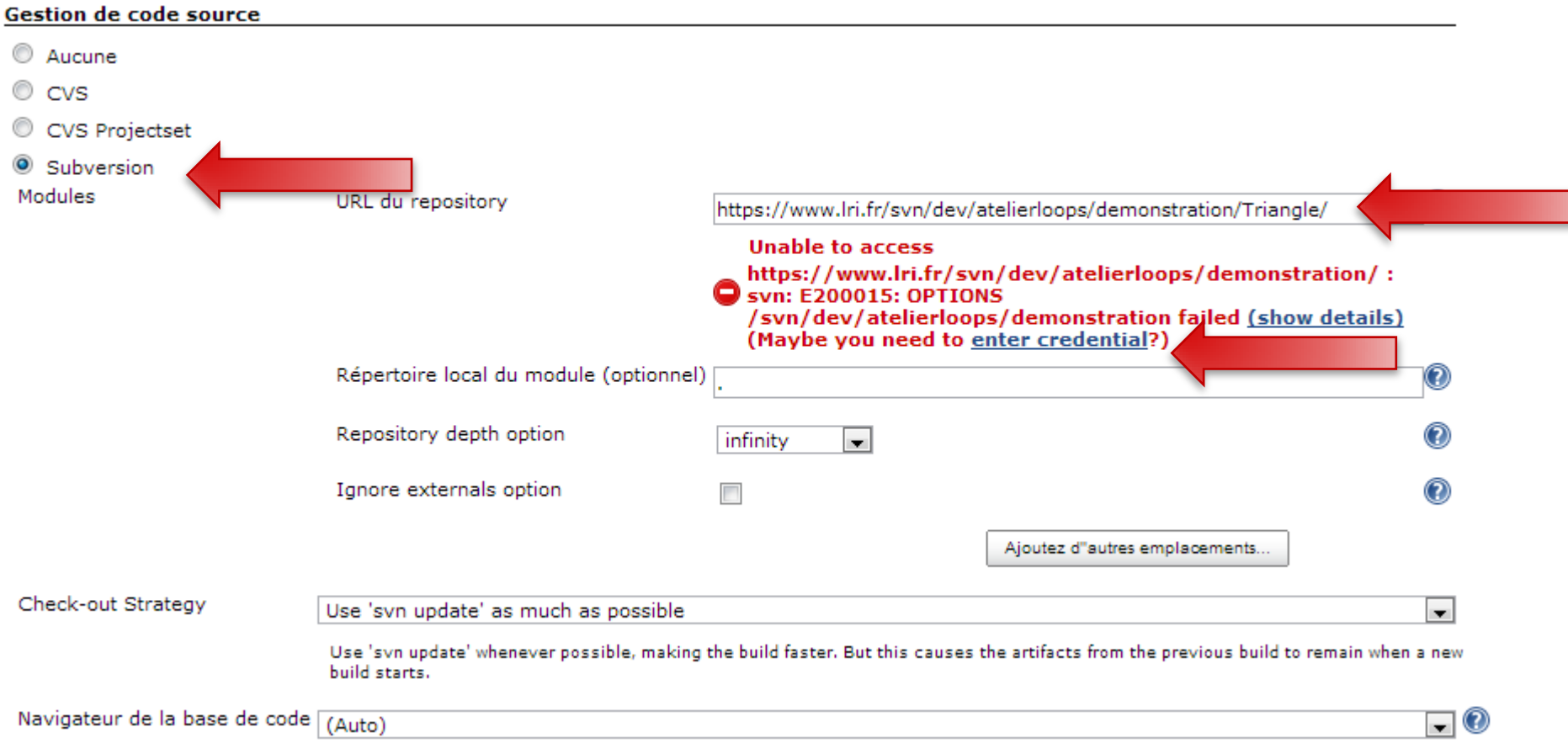


Création d'un nouveau job

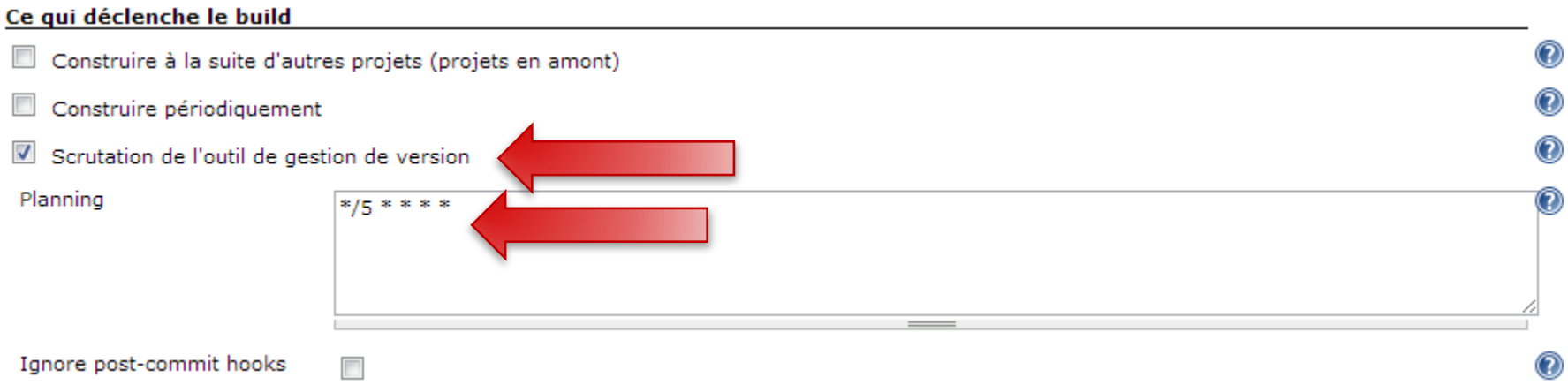


Configuration du job

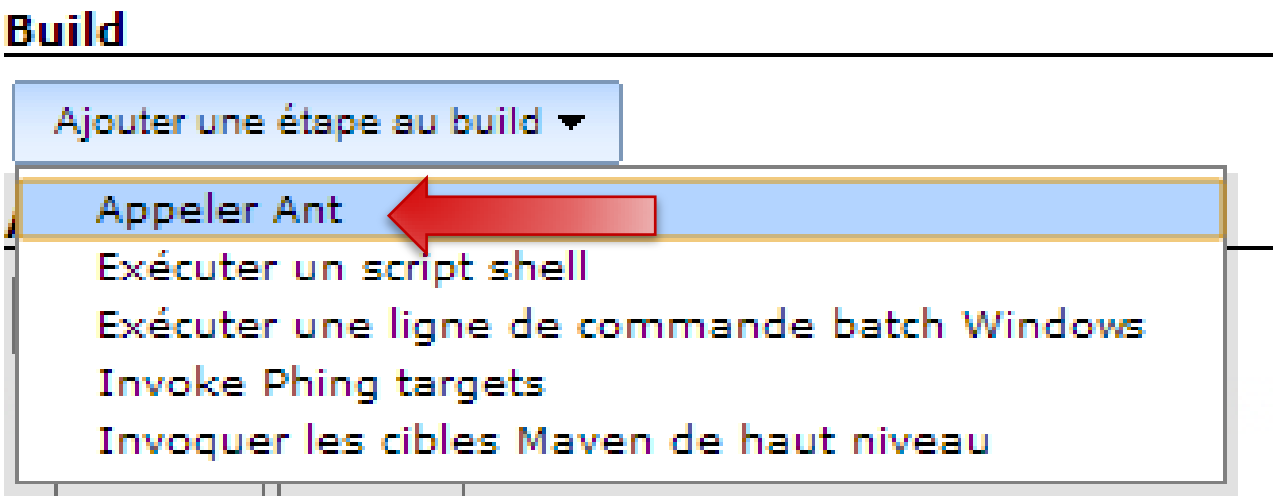
Configuration du dépôt

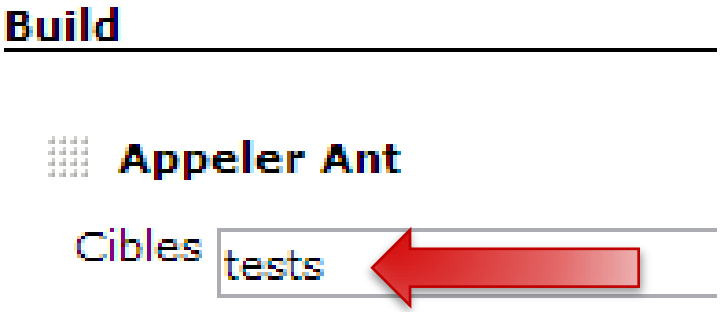


Build automatique

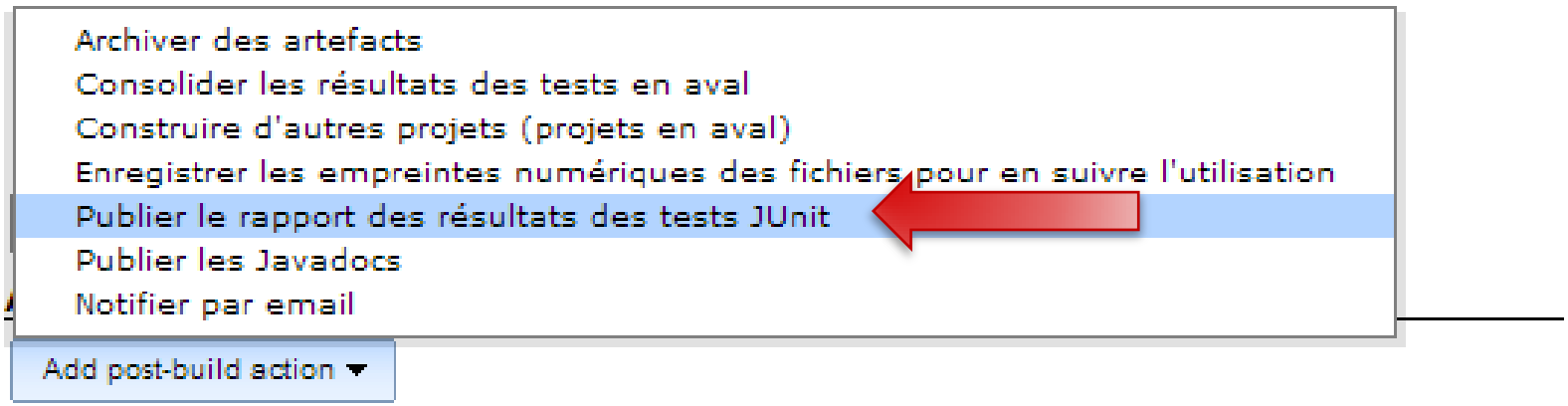


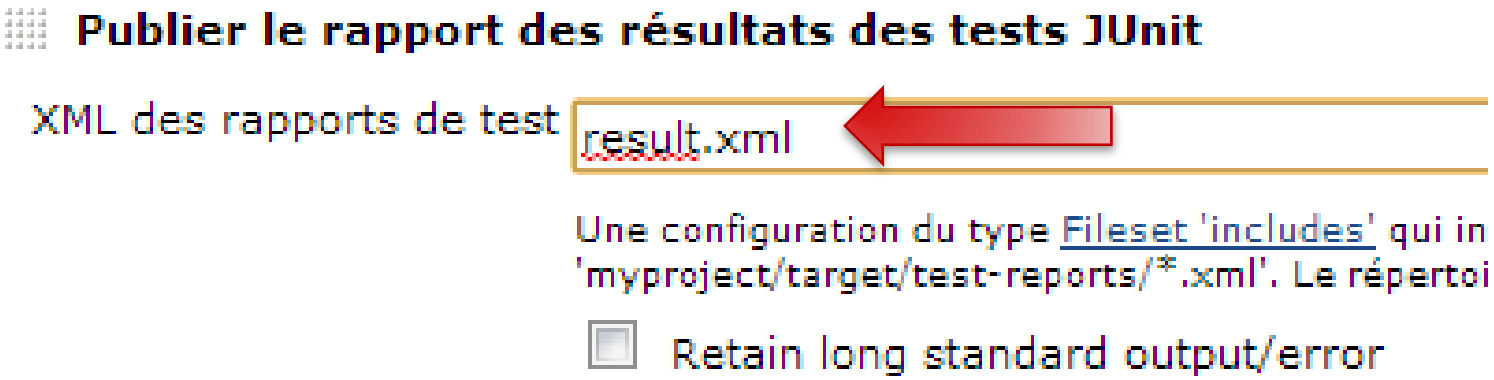
Ajout d'une étape de build





Ajout d'une étape post-build





Forcer l'exécution d'un build

Résultat des tests

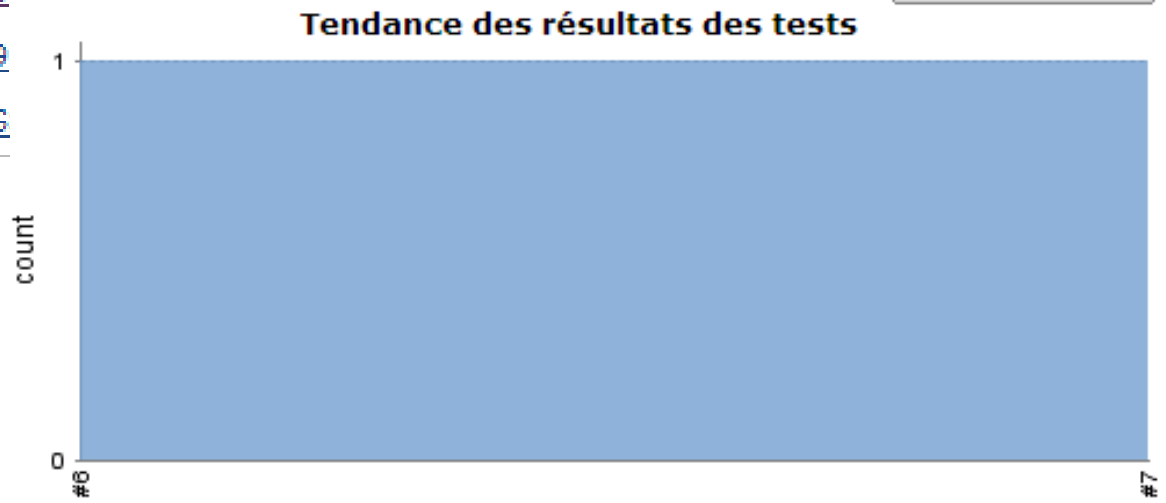
Historique des builds (Tendances)

#7	17 mars 2013 09:37:14
#6	17 mars 2013 09:36:45
#5	17 mars 2013 09:23:43
#4	17 mars 2013 09:20:16
#3	16 mars 2013 23:57:59
#2	16 mars 2013 23:55:23
#1	16 mars 2013 23:35:29

RSS toutes les constructions RS

ajouter une description

Désactiver le Projet



(montrer les échecs seulement) agrandir

Pour aller plus loin

- Intégration du code coverage (p.ex. Cobertura)
- Analyse statique de code
  - https://wiki.jenkins-ci.org/display/JENKINS/Static+Code+Analysis+Plug-ins
- Ajouter et configurer un esclave
- Configuration de tâches multiples

Exercice

- Ecrivez une propriété isRight() qui renvoie vrai ssi le triangle est rectangle
  - (3, 4, 5) est rectangle
- Testez
- Intégrez

Qu’avons-nous vu ?

1. Création d’un projet simple en Java
2. Rédaction de tests avec Junit 3
3. Couverture de code
4. Automatisation des tests
5. Intégration continue