

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

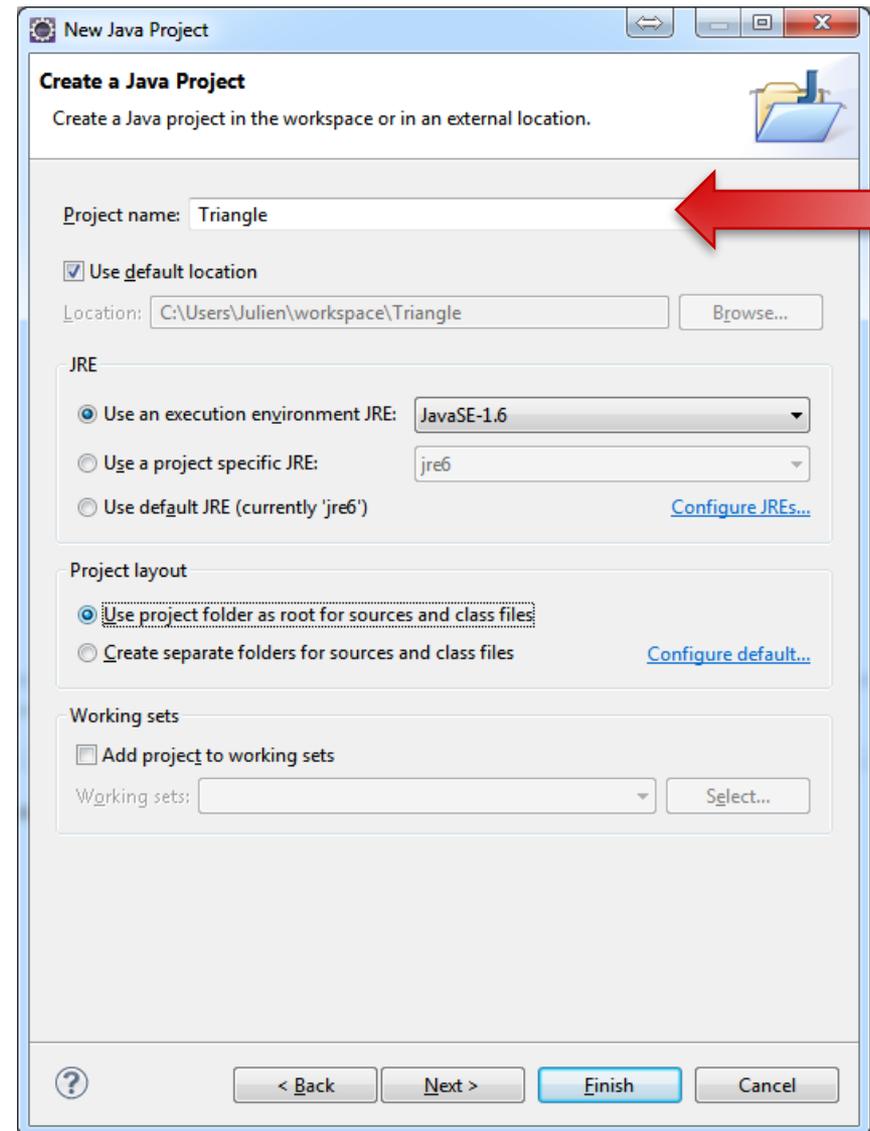
- Installer Java 1.6 – 1.7
- Installer Ant
- Installer Eclipse Juno 4.2
- Télécharger Jenkins (<http://jenkins-ci.org/>)

Utilisateurs de Windows: mettez à jour les variables système

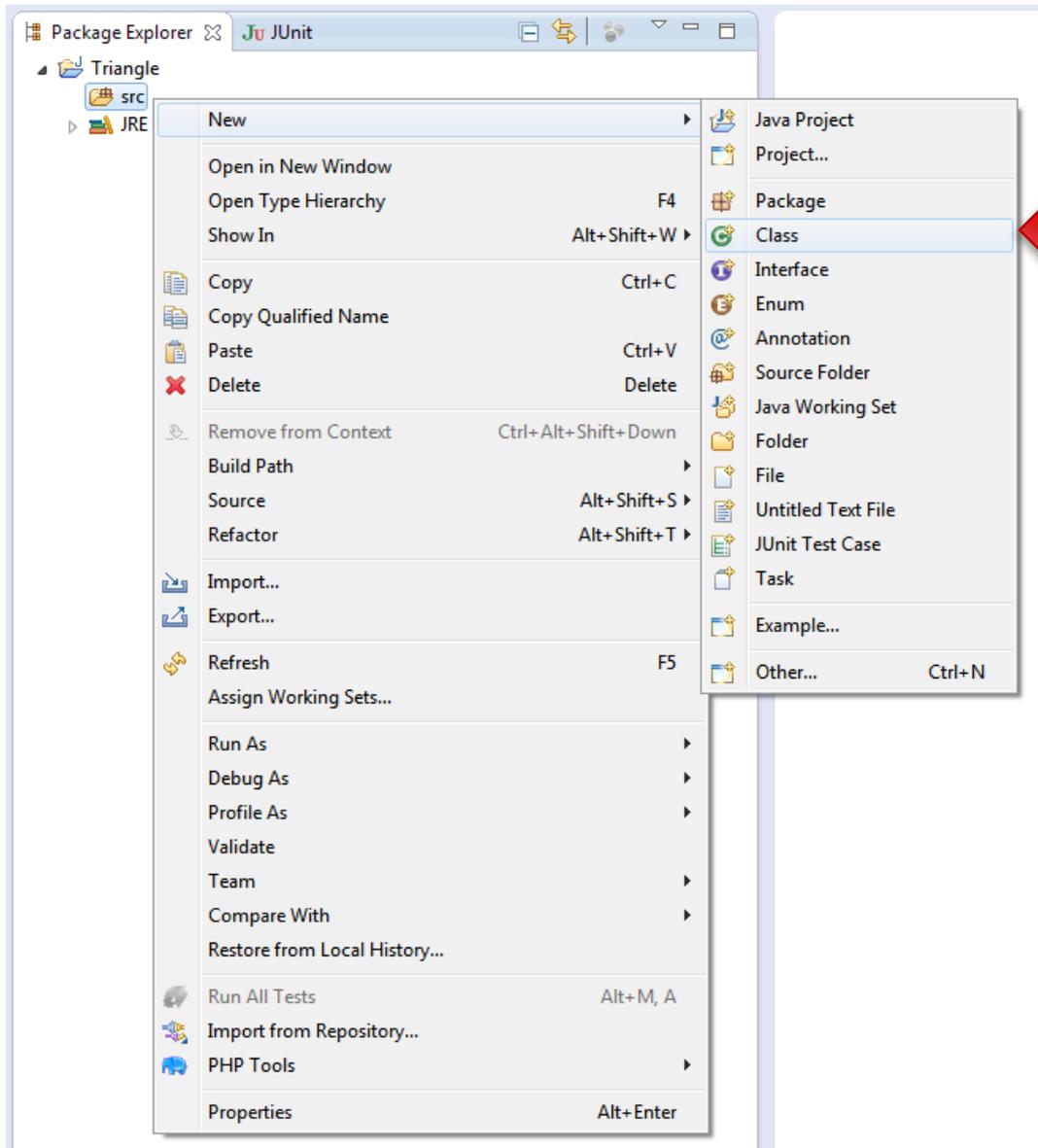
1. Création d'un projet simple

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:

Package:

Enclosing type:

Name: ←

Modifiers: public default private protected
 abstract final static

Superclass:

Interfaces:

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

Classe Triangle

New Java Class

Java Class
⚠ The use of the default package is discouraged.

Source folder: Triangle/src

Package: (default)

Enclosing type:

Name: Triangle

Modifiers: public default private protected
 abstract final static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

public static void main(String[] args)

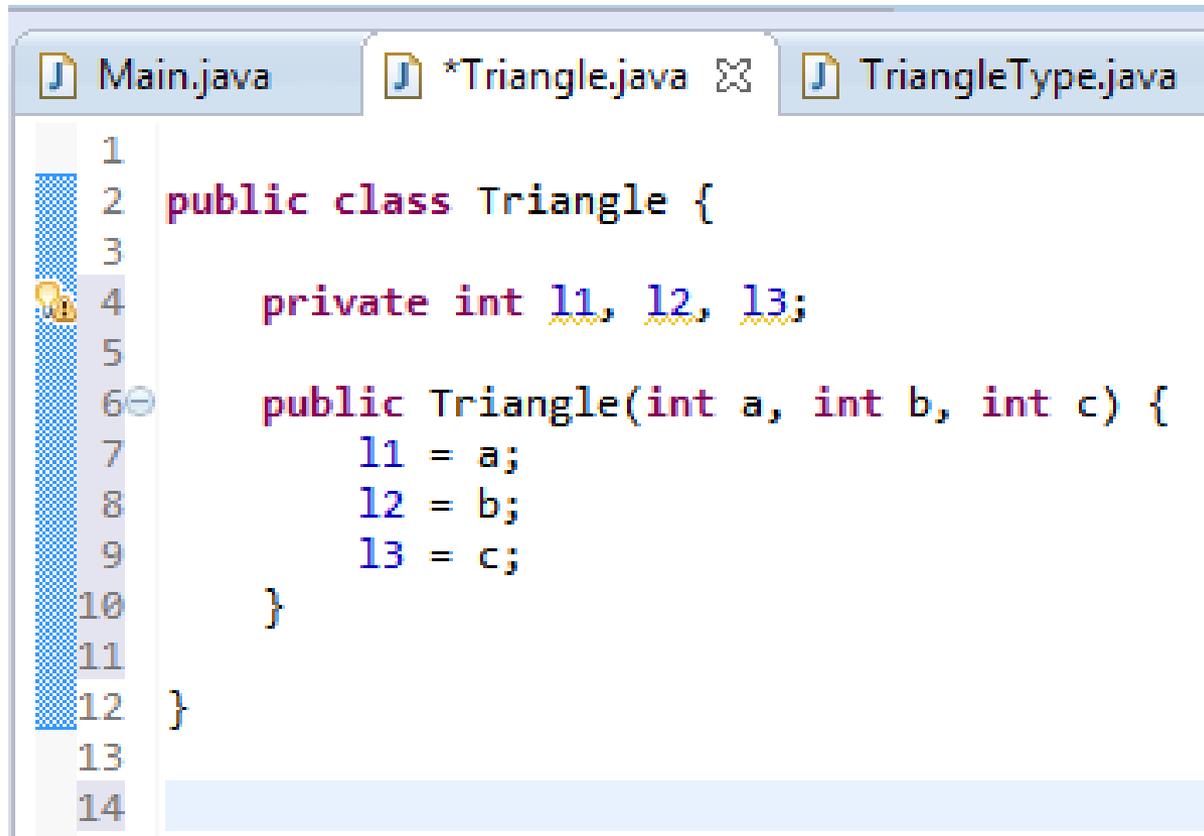
Constructors from superclass

Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))

Generate comments

Contenu de Triangle



```
1  
2 public class Triangle {  
3  
4     private int l1, l2, l3;  
5  
6     public Triangle(int a, int b, int c) {  
7         l1 = a;  
8         l2 = b;  
9         l3 = c;  
10    }  
11  
12 }  
13  
14
```

Enumération TriangleType

Enum Type

⚠ The use of the default package is discouraged.

Source folder: Triangle/src Browse...

Package: (default) Browse...

Enclosing type: Browse...

Name: TriangleType

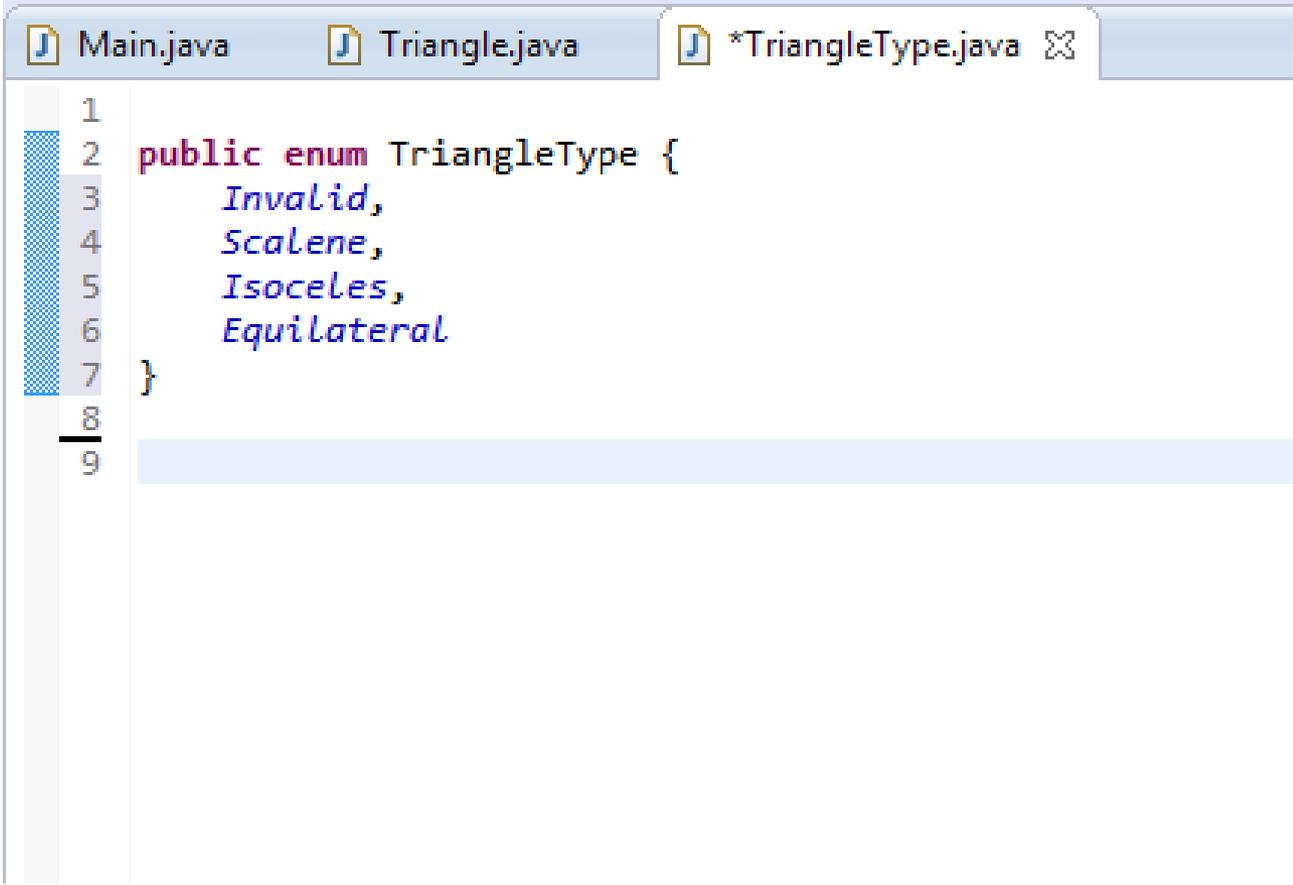
Modifiers: public default private protected

Interfaces: Add...
Remove

Do you want to add comments? (Configure templates and default value [here](#))
 Generate comments

? Finish Cancel

Contenu de TriangleType



```
1  
2 public enum TriangleType {  
3     Invalid,  
4     Scalene,  
5     Isoceles,  
6     Equilateral  
7 }  
8  
9
```

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 :

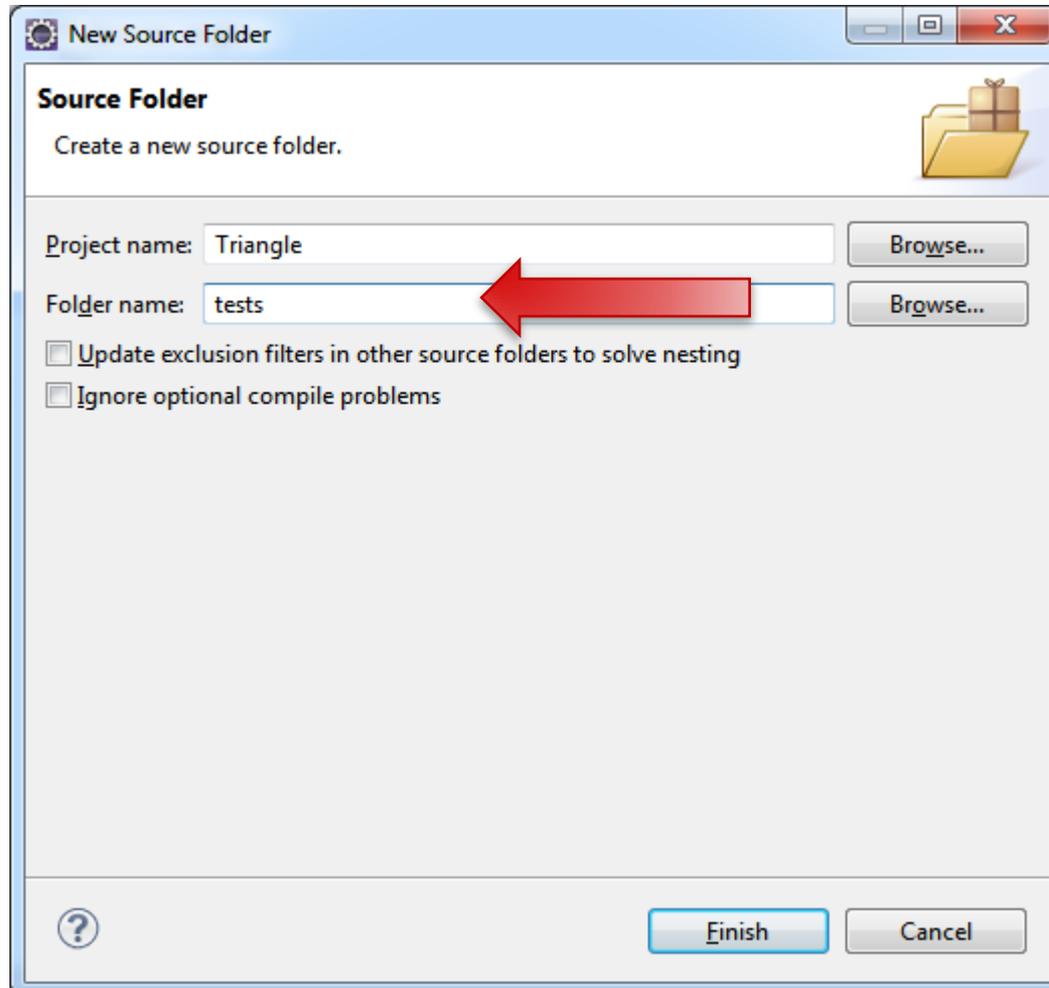
```
public static void main(String[] args) {  
    Triangle t1 = new Triangle(3, 4, 5);  
    System.out.println("Triangle t1 is " + t1.getType());  
  
    Triangle t2 = new Triangle(0, 0, 0);  
    System.out.println("Triangle t2 is " + t2.getType());  
  
    Triangle t3 = new Triangle(2, 3, 7);  
    System.out.println("Triangle t3 is " + t3.getType());  
}
```

- Exécutez. Le résultat est-il correct ?

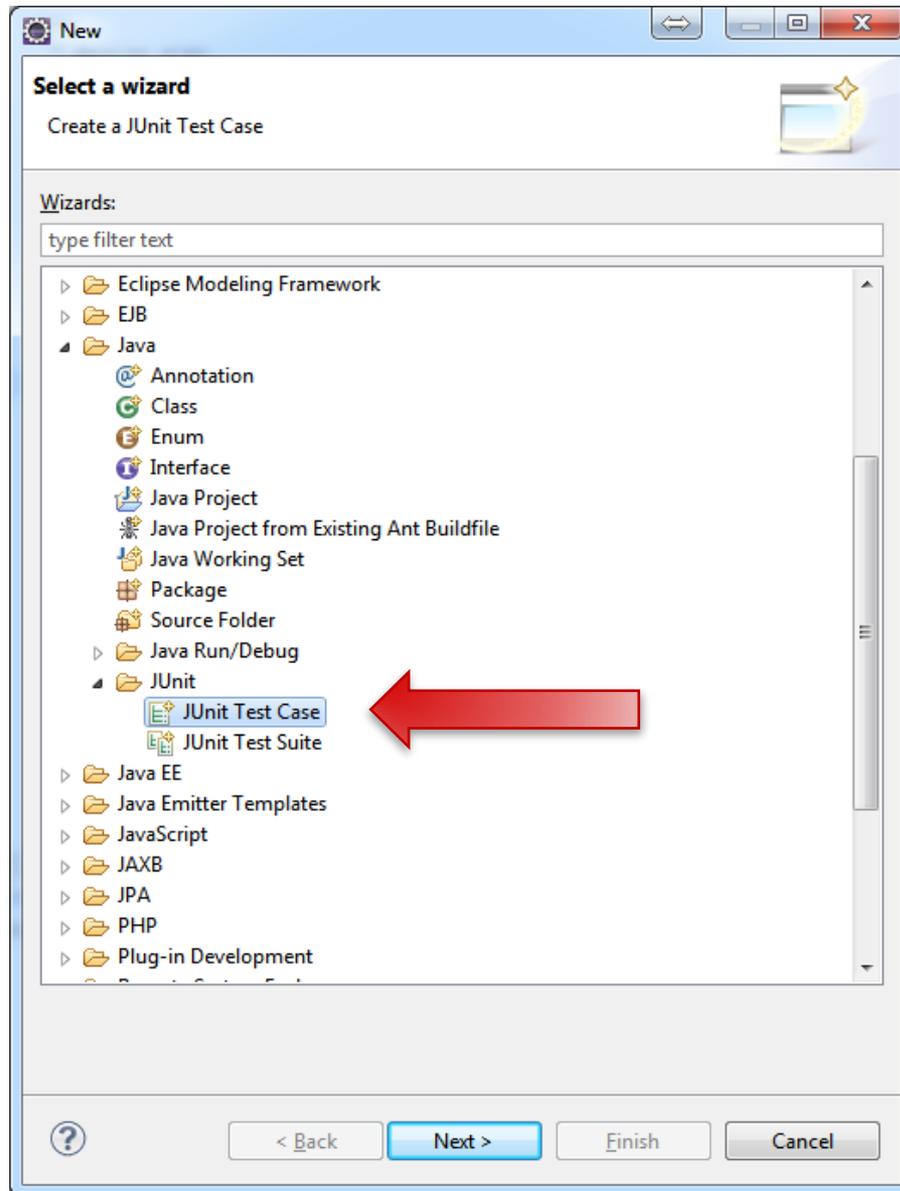
2. Rédaction de tests avec JUnit

Création d'un dossier de tests

- New -> Source Folder



Création d'un test



Paramètres du test

New JUnit Test Case

JUnit Test Case

⚠ The use of the default package is discouraged.

New JUnit 3 test New JUnit 4 test

Source folder:

Package:

Name:

Superclass:

Which method stubs would you like to create?

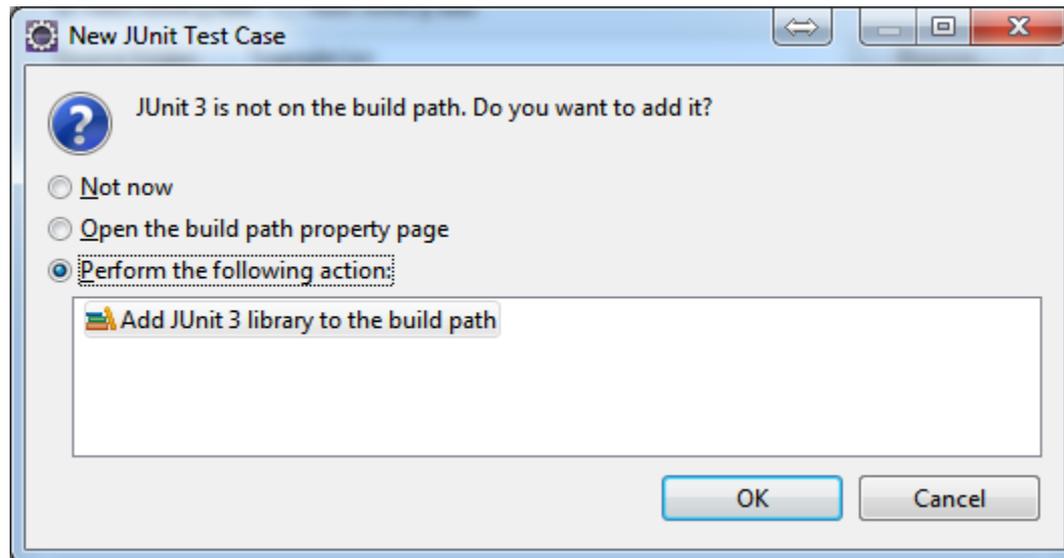
setUpBeforeClass() tearDownAfterClass()
 setUp() tearDown()
 constructor

Do you want to add comments? (Configure templates and default value [here](#))

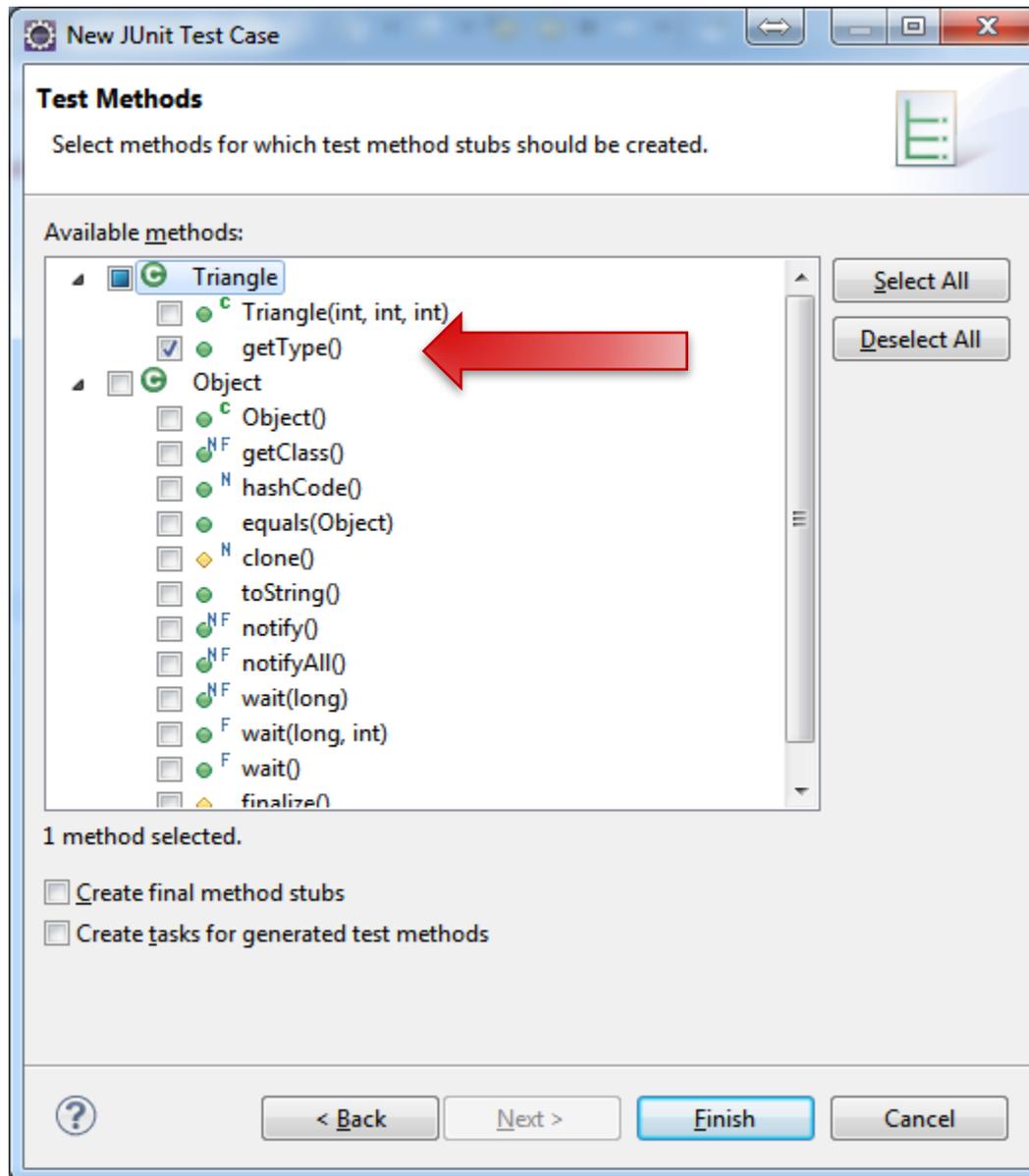
Generate comments

Class under 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();  
    }
```

⊖

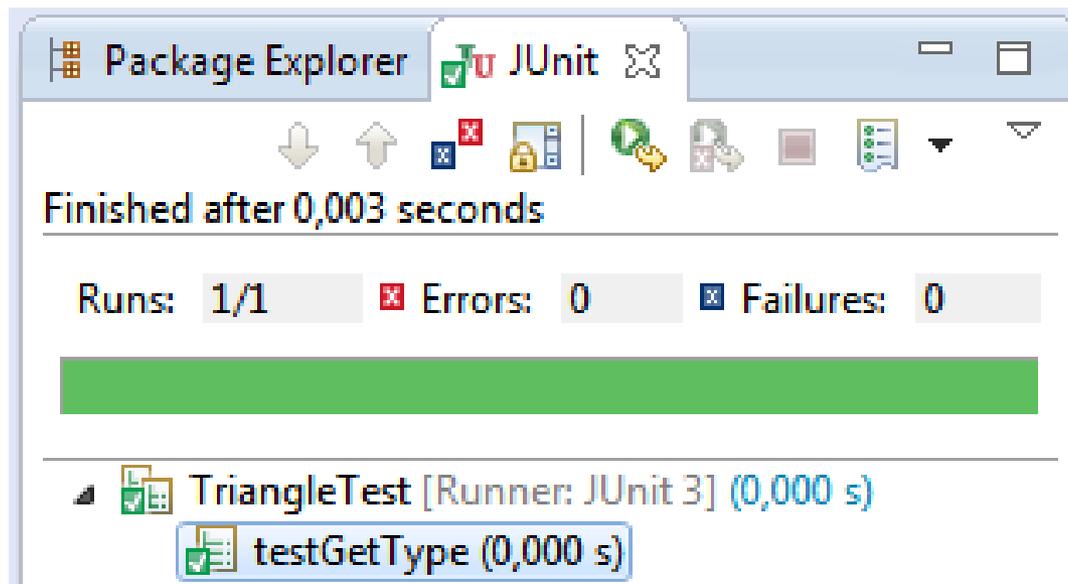
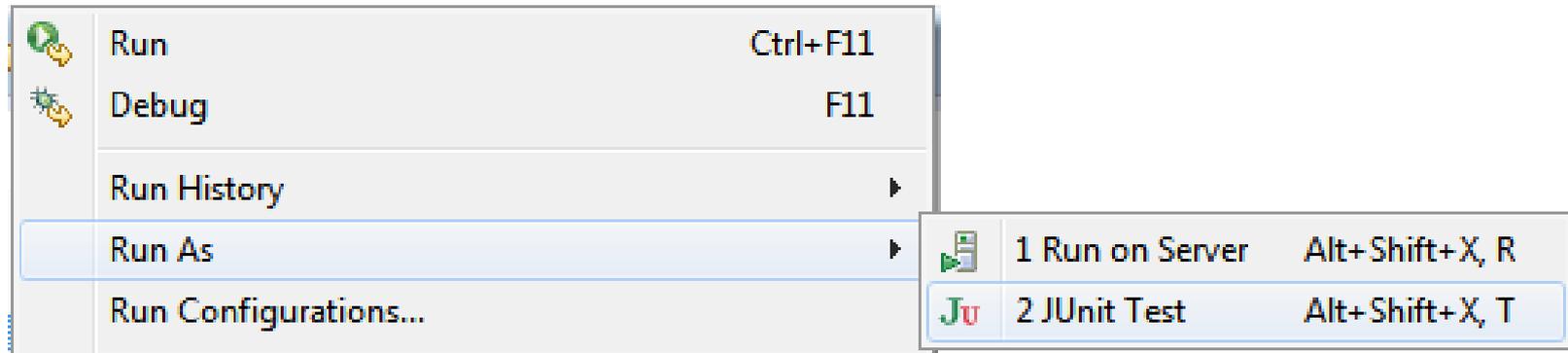
```
    public void testGetType() {  
        fail("Not yet implemented");  
    }
```

```
}
```

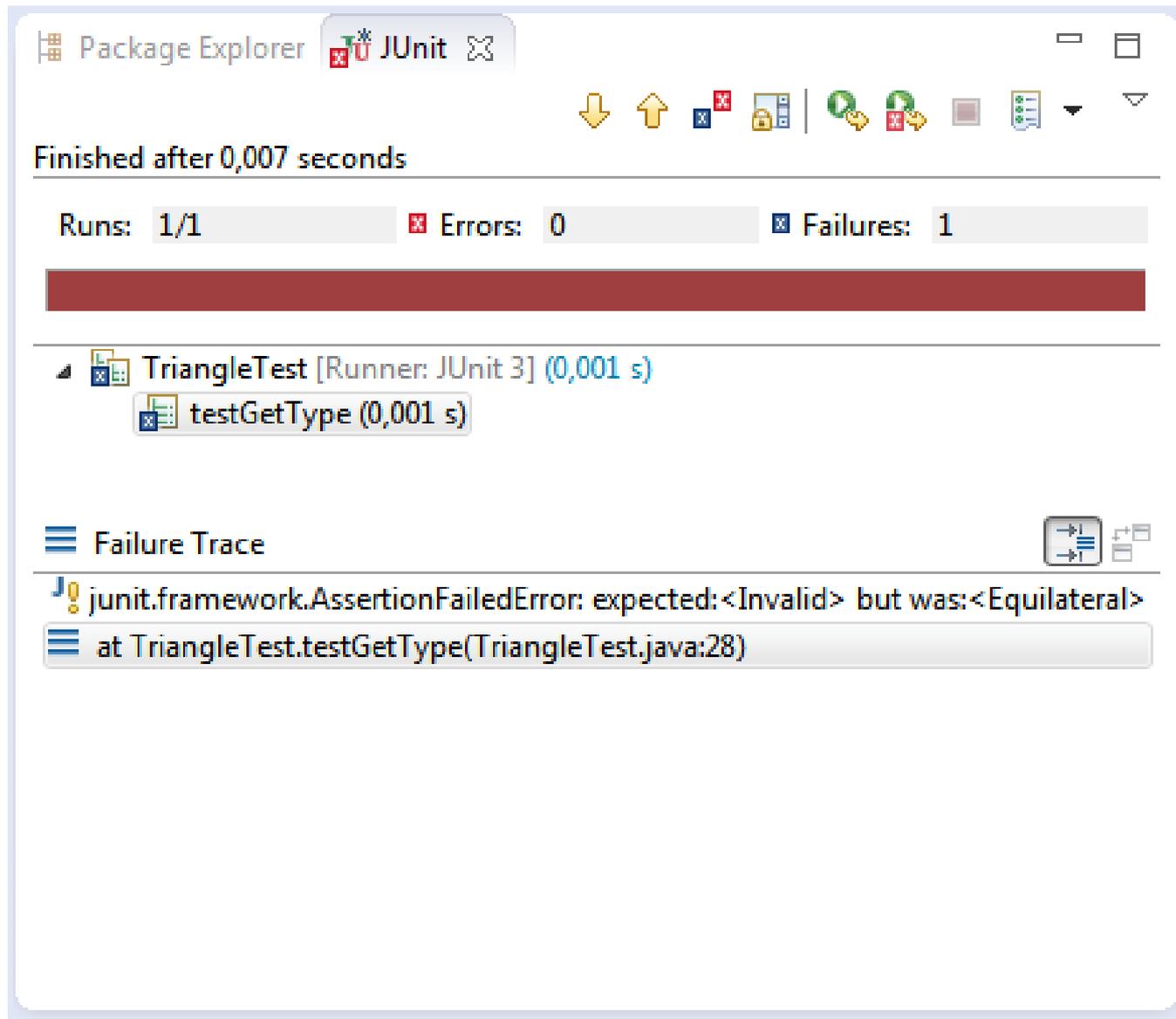
Un premier test

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

Exécuter les tests



Et si on se trompe ?



The screenshot shows the JUnit runner interface in an IDE. At the top, it says "Package Explorer" and "JUnit". Below that, it indicates "Finished after 0,007 seconds". The summary shows "Runs: 1/1", "Errors: 0", and "Failures: 1". A red progress bar is visible. The test results list "TriangleTest [Runner: JUnit 3] (0,001 s)" and "testGetType (0,001 s)". The failure trace shows the error message: "junit.framework.AssertionFailedError: expected:<Invalid> but was:<Equilateral>" and the location "at TriangleTest.testGetType(TriangleTest.java:28)".

Package Explorer JUnit

Finished after 0,007 seconds

Runs: 1/1 Errors: 0 Failures: 1

TriangleTest [Runner: JUnit 3] (0,001 s)

testGetType (0,001 s)

Failure Trace

junit.framework.AssertionFailedError: expected:<Invalid> but was:<Equilateral>
at TriangleTest.testGetType(TriangleTest.java:28)

Exercice

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

Pour aller plus loin...

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

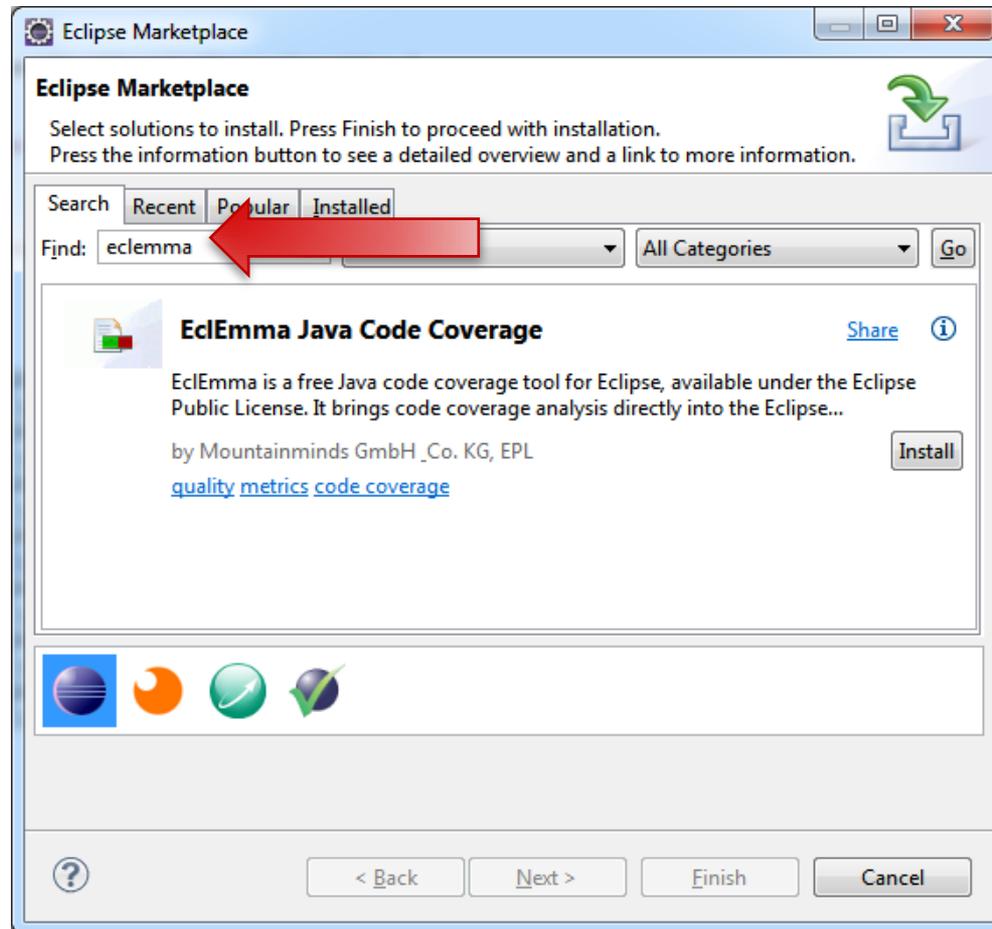
Pour aller plus loin...

- 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

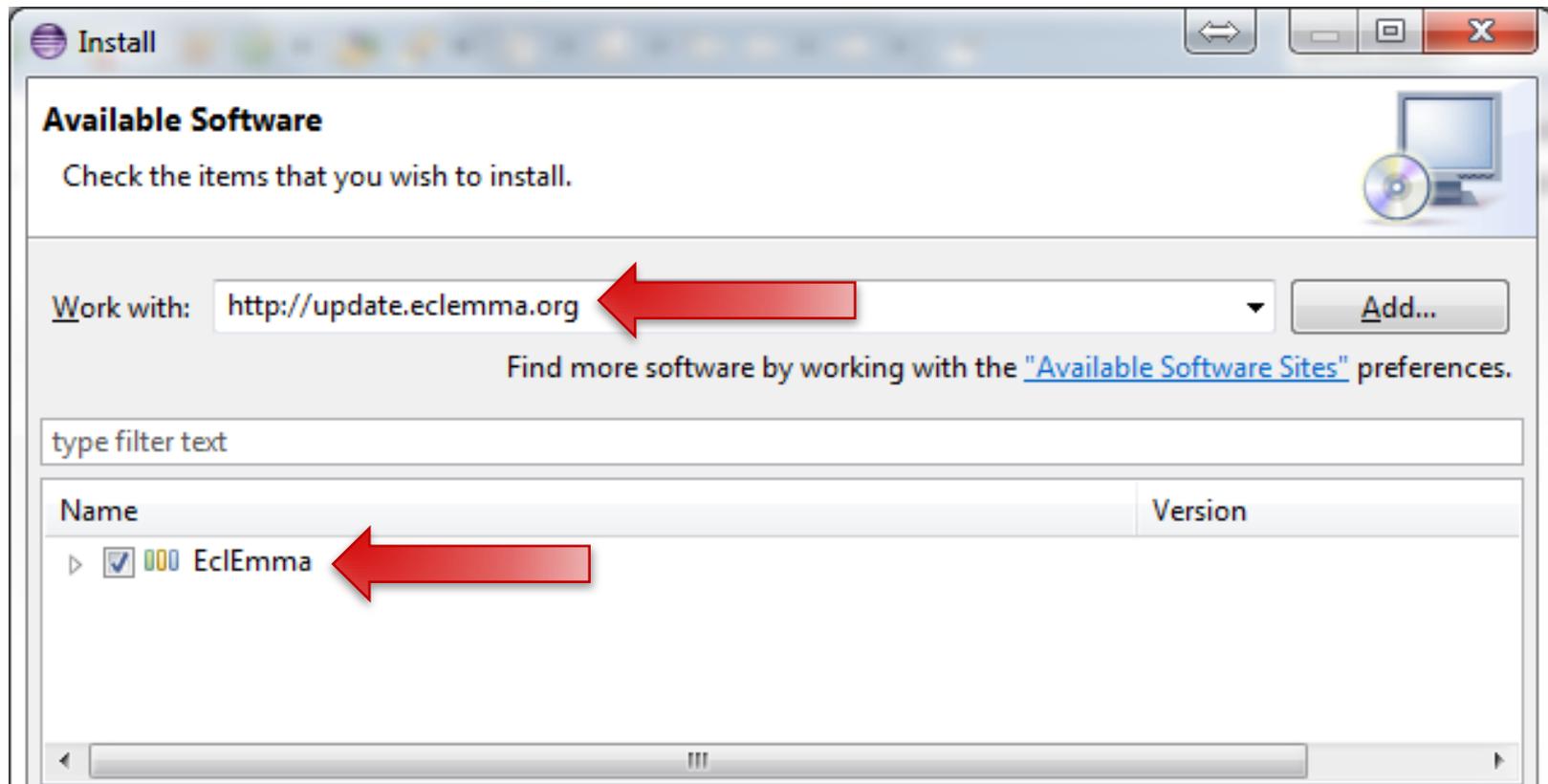
Installation d'EclEMMA

- Help -> Eclipse Marketplace

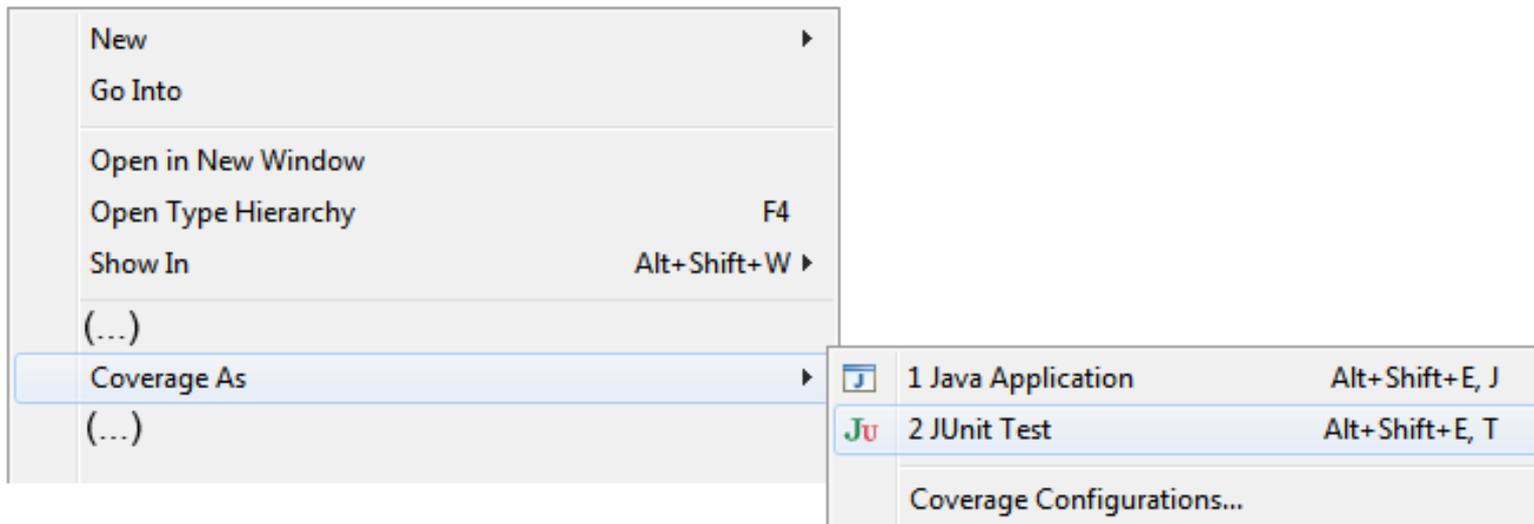


Installation d'EclEmma (2)

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



Utilisation d'EcLEmma



Résultat

```
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 sceanele 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;
}
```

```
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. Automatisierung

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

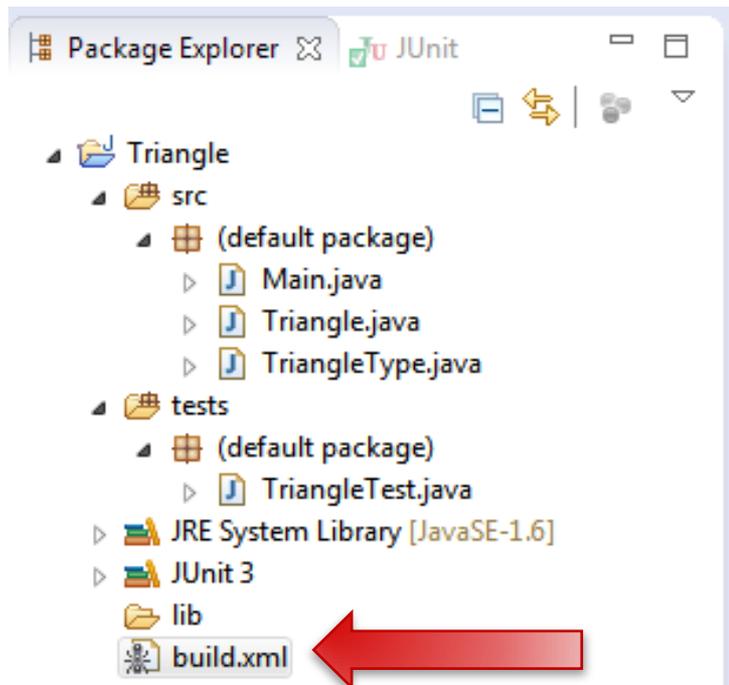
OU

- 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



```
build.xml
```

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>  
<project>  
  
</project>
```

Création de cibles

build.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



The screenshot shows the Jenkins web interface in a browser window. The browser tab is titled "Tableau de bord [Jenkins]" and the address bar shows "localhost:8080". The Jenkins logo is prominently displayed at the top. Below the logo, there are several navigation links: "Nouveau Job", "Utilisateurs", "Historique des constructions", and "Administrer Jenkins". A welcome message reads: "Bienvenue sur Jenkins ! Veuillez [créer un nouveau job](#) pour démarrer." Below this, there is a section titled "File d'attente des constructions" which shows a table with two rows, both in the "En attente" (Waiting) status. At the bottom of the page, there is a link to "Aidez-nous à traduire cette page".

Tableau de bord [Jenkins] x

localhost:8080

Jenkins

Jenkins

[Nouveau Job](#)

[Utilisateurs](#)

[Historique des constructions](#)

[Administrer Jenkins](#)

Bienvenue sur Jenkins ! Veuillez [créer un nouveau job](#) pour démarrer.

File d'attente des constructions

File d'attente des constructions vide

État du lanceur de constructions

#	Statut
1	En attente
2	En attente

[Aidez-nous à traduire cette page](#)

Création d'un nouveau job

Nom du Job

Construire un projet free-style

Ceci est la fonction principale de Hudson qui sert à builder (construire) votre projet. Vous pouvez intégrer tous les outils de gestion de version avec tous les systèmes de build. Il est même possible d'utiliser Hudson pour tout autre chose qu'un build logiciel.

Construire un projet maven2/3

Construit un projet avec maven2/3. Jenkins utilise directement vos fichiers POM et diminue radicalement l'effort de configuration.

Construire un projet multi-configuration

Adapté aux projets qui nécessitent un grand nombre de configurations différentes, comme des environnements de test multi

Contrôler un job externe

Ce type de tâche permet de suivre l'exécution d'un process lancé en dehors de Hudson, même sur une machine distante. Cette option permet l'utilisation de Hudson comme tableau de bord de votre environnement d'automatisation. Voir [la documentation](#) pour plus de détails.

Copier un Job existant

Copier à partir de

OK

Configuration du job

Jenkins

Jenkins > Atelier Intégration Continue

 [Retour au tableau de bord](#)

 [État](#)

 [Modifications](#)

 [Espace de travail](#)

 [Lancer un build](#)

 [Supprimer ce Projet](#)

 [Configurer](#)



Historique des builds

([Tendances](#))

 [RSS toutes les constructions](#)  [RSS de tous les échecs](#)

Projet Atelier Intégration Continue



[Espace de travail](#)



[Changements récents](#)

Liens permanents

Configuration du dépôt

Gestion de code source

- Aucune
 - CVS
 - CVS Projectset
 - Subversion
- Modules

URL du repository

Unable to access

https://www.lri.fr/svn/dev/atelierloops/demonstration/ :
svn: E200015: OPTIONS
/svn/dev/atelierloops/demonstration failed ([show details](#))
(Maybe you need to [enter credential?](#))

Répertoire local du module (optionnel)



Repository depth option



Ignore externals option



Check-out Strategy



Use 'svn update' whenever possible, making the build faster. But this causes the artifacts from the previous build to remain when a new build starts.

Navigateur de la base de code



Build automatique

Ce qui déclenche le build

- Construire à la suite d'autres projets (projets en amont)
- Construire périodiquement
- Scrutation de l'outil de gestion de version

Planning

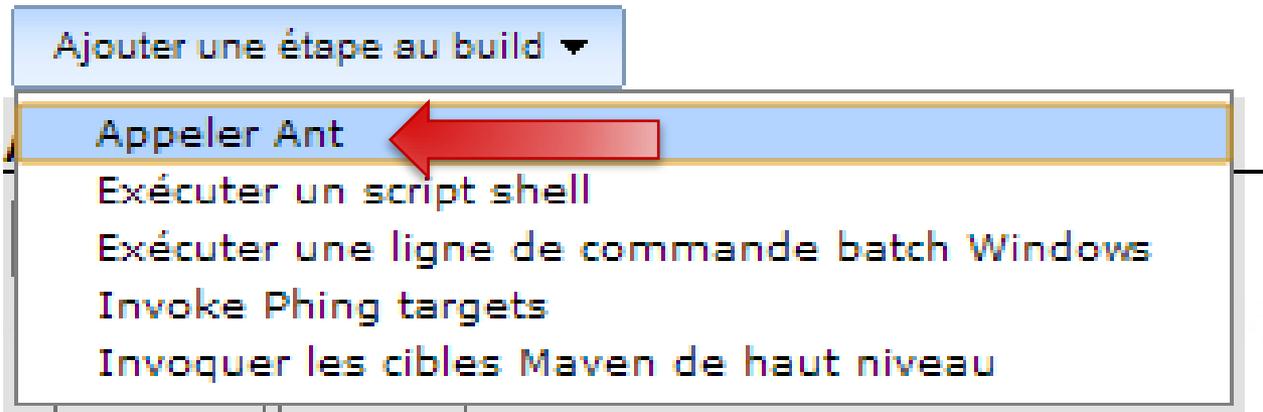
```
*/5 * * * *
```

Ignore post-commit hooks



Ajout d'une étape de build

Build



Build

Appeler Ant

Cibles A red arrow points to the "tests" text in the input field.

Ajout d'une étape post-build

Archiver des artefacts

Consolider les résultats des tests en aval

Construire d'autres projets (projets en aval)

Enregistrer les empreintes numériques des fichiers pour en suivre l'utilisation

Publier le rapport des résultats des tests JUnit

Publier les Javadocs

Notifier par email

Add post-build action ▼

Publier le rapport des résultats des tests JUnit

XML des rapports de test

Une configuration du type Fileset 'includes' qui in
'myproject/target/test-reports/*.xml'. Le réperto

Retain long standard output/error

Forcer l'exécution d'un build

Jenkins

Jenkins > Atelier Intégration Continue

-  [Retour au tableau de bord](#)
-  [État](#)
-  [Modifications](#)
-  [Espace de travail](#)
-  [Lancer un build](#)
-  [Supprimer ce Projet](#)
-  [Configurer](#)
-  [Log du dernier accès à Subversion](#)

 Historique des builds	(Tendances)
 #2	16 mars 2013 23:55:23
 #1	16 mars 2013 23:35:29
 RSS toutes les constructions	 RSS de tous les échecs

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

Tendance des résultats des tests



[\(montrer les échecs seulement\)](#) [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