Eclipse

1. Installation de Eclipse

Avant d'installer Eclipse, il faut le télécharger du site officiel : www.eclipse.org, onglet *download* (essayer de télécharger une version angalise et non française). Au bien vous le procurer à partir d'une autre personne (à l'université : vous pouvez le télécharger à partir de logithèque).

Il suffit de décompresser l'archive et avoir un dossier nommée généralement Eclipse (eclipse). Vous créer un raccourci vers *eclipse.exe* sur votre bureau (pour un lancement rapide).

En plus de ça, et pour que Eclipse fonctionne, vous devez aussi installer le JDK (Java Development Kit). Sinon, éclipse vous affiche un message d'erreur lors de son lancement.

Remarque que vous n'avez pas besoins de paramétrer ou de créer les variables environnements (**JAVA_PATH** pour trouver automatiquement les fichiers executable *javac.exe*, *java.exe*, et **CLASSES_PATH** pour la *bibliothèque de classes*, *etc.*). Eclipse cherche automatiquement l'endroit du JDK (ou bien du JRE).

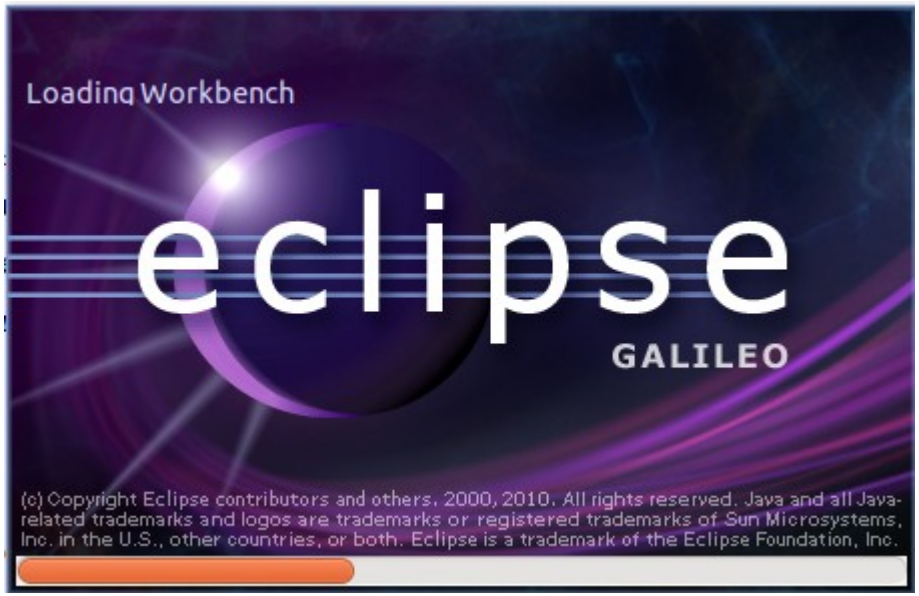


Fig1. Lancement du Eclipse

Pour la premier exécution de Eclipse, il y aura une boite de dialogue pour sélectionner ou choisir son ***workspace*** (espace de travail)

Le workspace est un emplacement dans le disque (chemin d'accès) qui indique l'endroit ou vos projets eclipse sont enregistrés. Vous pouvez le modifier à n'importe quel moment à travers la commande : File / Switch workspace / Other …

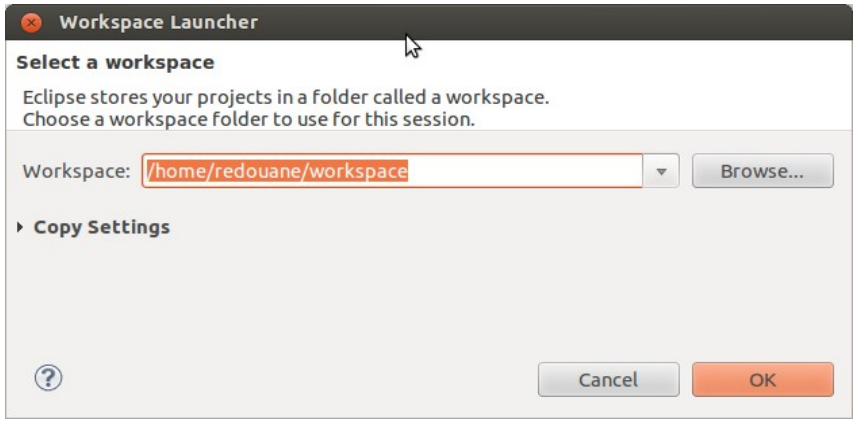


Fig2. Sélection du workspace

Eclipse affichera après une interface qui ressemble à la figure suivante :

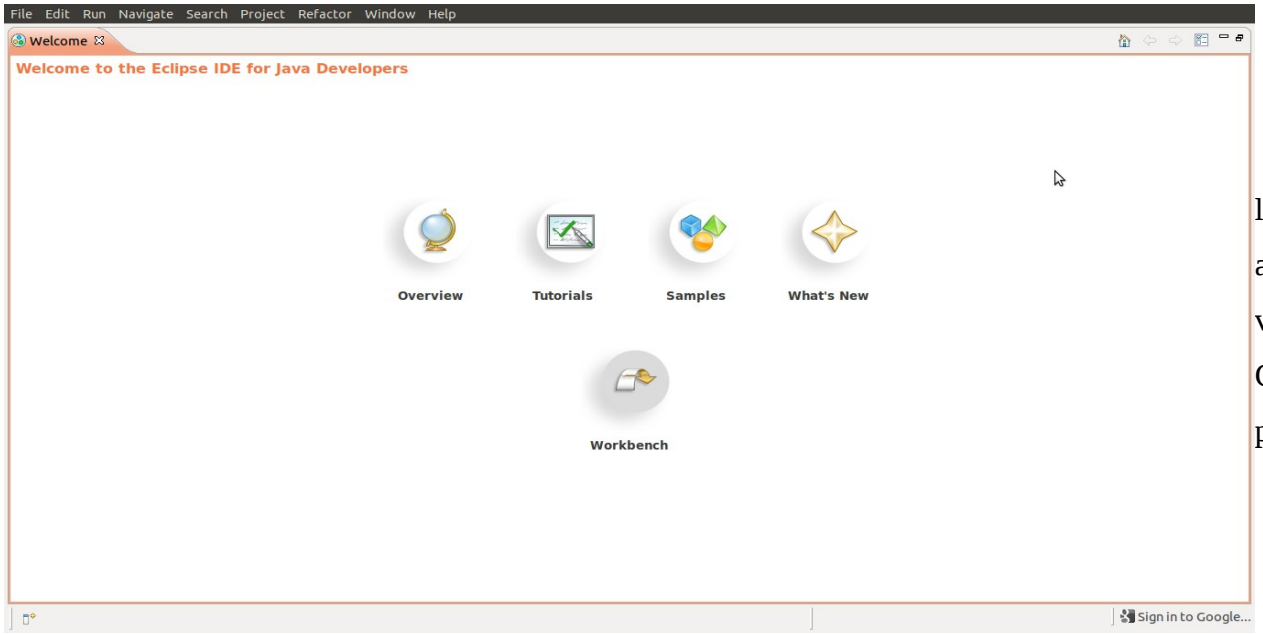


Fig3. La première interface du premier lancement de Eclipse

Bien évidemment, cette interface n'est pas l'interface de travail. Pour y accéder, il faut cliquer sur le bouton : 

.

Interface de travail

Le clic sur le bouton Workbench permet d'afficher l'interface de travail. Cette interface présente ce qui est appelé : Perspective. Une perspective c'est l'arrangement des vues montrées sur l'écran : Package Explorer, Outline, Console, Problems, Serarch, *etc*. Eclipse possède plusieurs perspectives. La figure suivante montre la perspective Java :

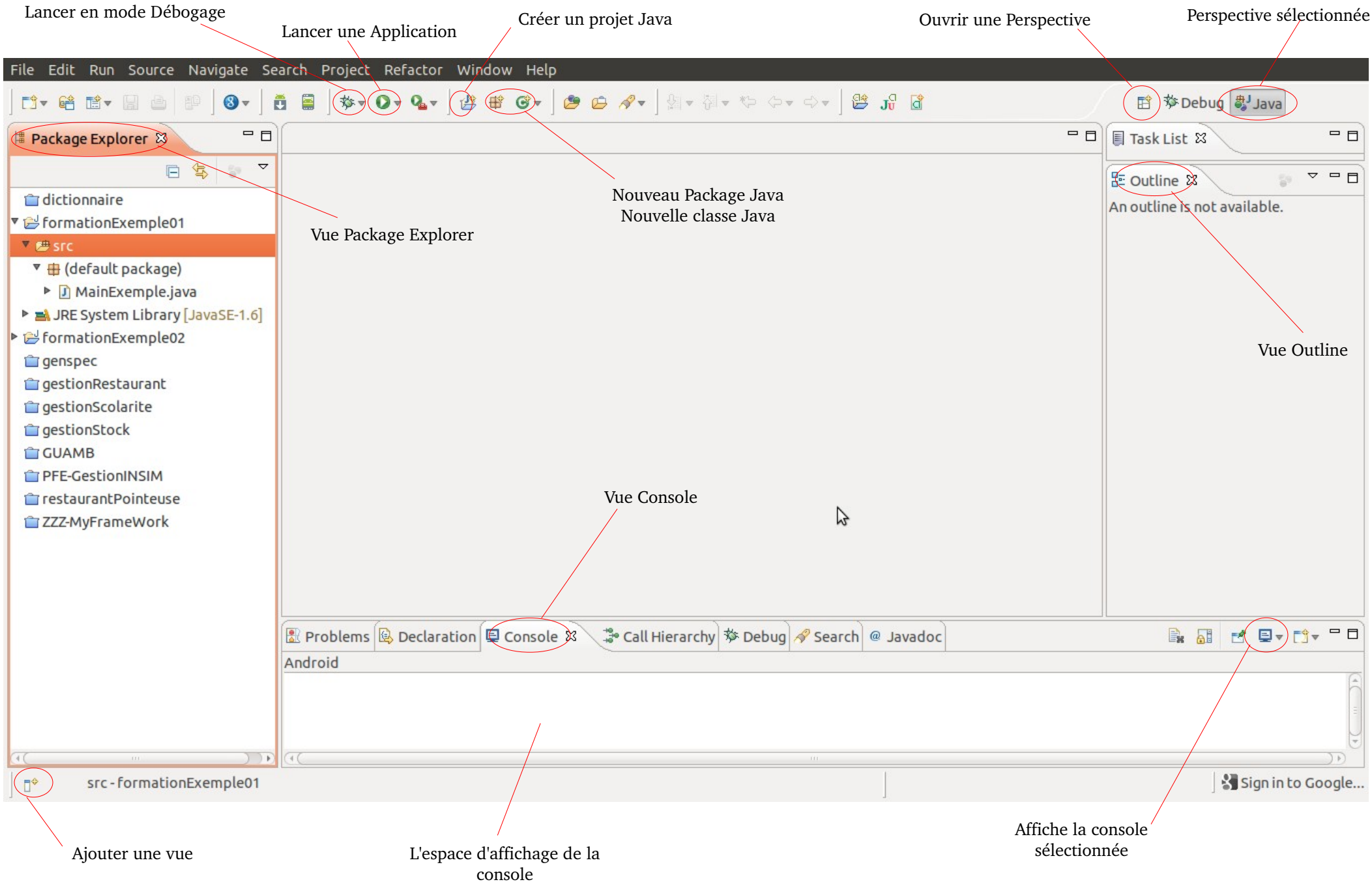


Fig4. Perspective Java

2. Créer un projet Java

Pour créer un projet Java, il suffit de cliquer sur le bouton  (ou bien via la commande File / New / Java Project). Il y aura l'affiche de la boite dialogue New Java Project :

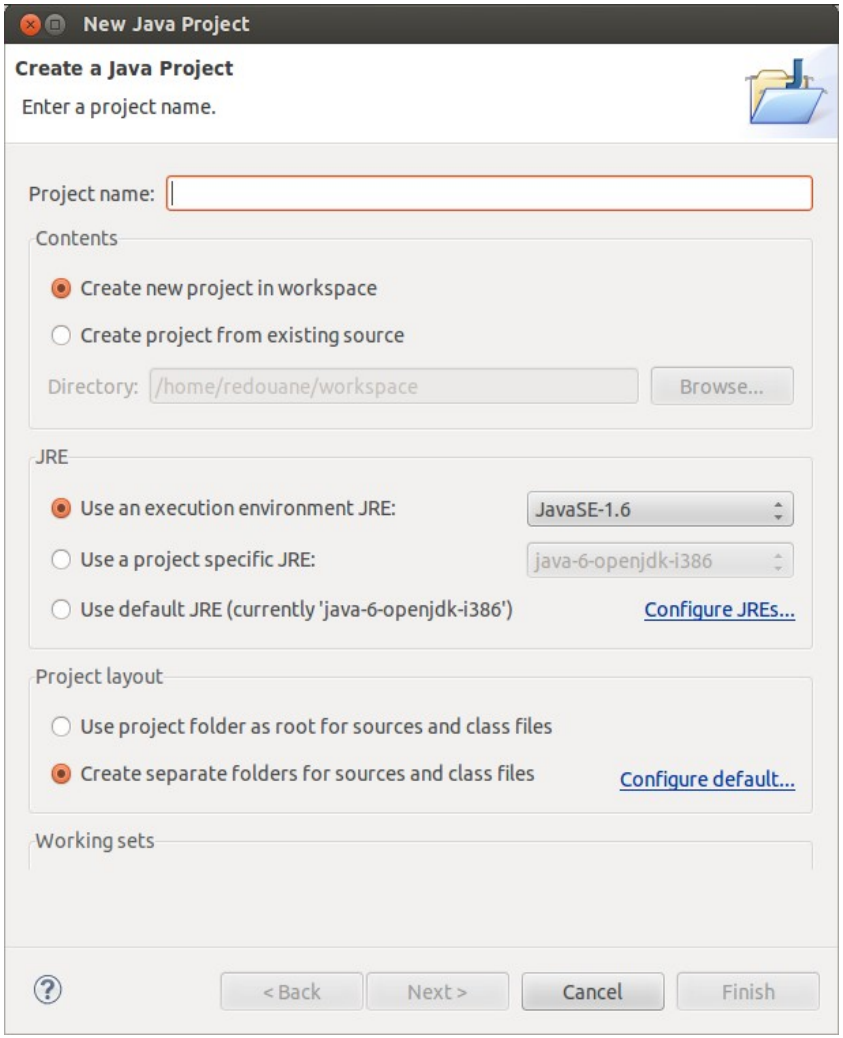
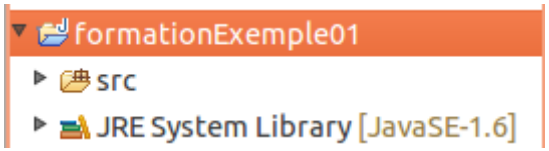


Fig5. Boite de dialogue Nouveau Projet Java

Dans cette boite de dialogue, vous indiquez principalement le nom du projet. On clique ensuite sur *Finish*.

Dans le cas où on veut créer un projet à partir d'un projet existant et non vide (un dossier mis dans le workspace) on clique sur le bouton *Next>*.

Un nouveau projet vide contient les élément suivant :



Le plus important est le dossier *src* qui contient les fichiers sources (*.java) de votre projet. Tous les fichiers Java (classes Java) seront créer dans ce dossier.

La compilation et vérification des erreurs syntaxiques se fait d'une manière transparente et automatique. Pour chaque fichier Java compilé, Eclipse génère un fichier .class dans le répertoir bin (qui n'est pas affiché sur l'interface de Eclipse).

Remarque :

Il est recommander de mettre les fichiers java dans des packages dans le dossier *src*.

La prochaine étape est de créer une classe (cependant, dans la pratique et pour une bonne programmation, on crée tout d'abord des packages afin de structurer les classes). Pour créer

une classe, on clique sur le bouton . (ou bien File / New / Class).

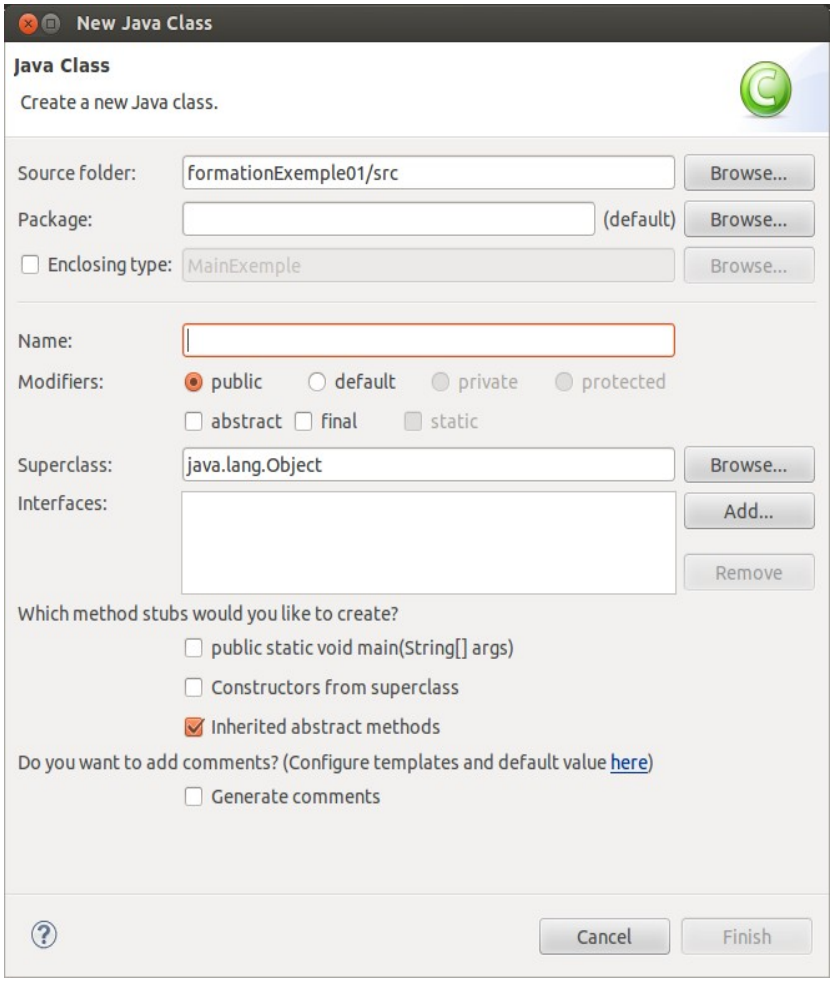
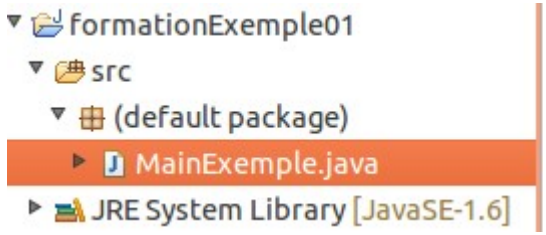
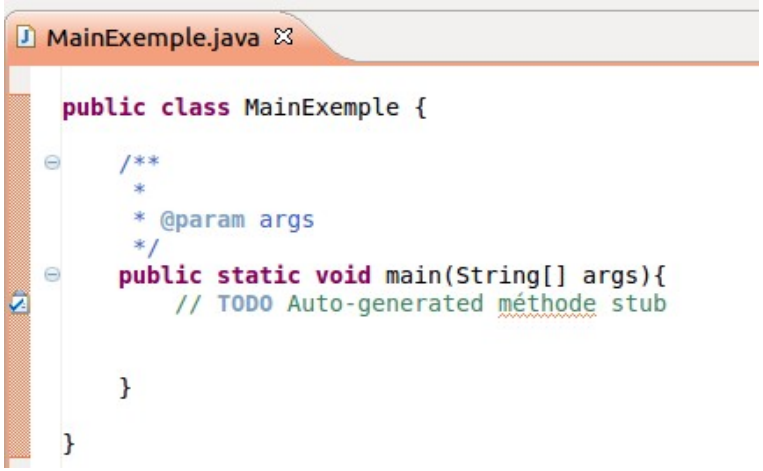


Fig6. Boite de Dialogue Nouvelle Classe Java

Dans la boite de dialogue New Java Class, on indique le nom de la classe (comme identificateur : pas d'espace, pas de symbole, il doit commencer par un caractère alphabétique, etc.) On peut cocher la case *public static void main(String[] args)* pour créer automatiquement la méthode principale. Si on créer une classe nommée : MainExemple on aura l'affichage suivant :



(Eclipse ajoute automatiquement l'extension java au nom de la classe).



Remarque :

Pour écrire des commentaires sur Eclipse, on peut utiliser :

- /* … … … … */ : pour les commentaires multi-lignes
- // : pour les commentaires d'une seule ligne de code
- /** …. … … … */ : pour les commentaires JavaDoc

L'utilisation de // TODO permet à Eclipse d'identifier l'endroit par l'indication afin de la localiser rapidement. Ceci permet de marquer des morceaux de code en chantier (surtout dans un code volumineux).

Au fur et à mesure qu'on tape du code, on remarque :

- Si on marque une petite pause après un point tapé, éclipse affiche une liste contenant toutes les méthodes / attributs que vous pouvez utiliser.
- Si on mis le curseur au dessus d'un attributs ou méthode, éclipse affiche la documentation correspondante.
- Si on fait des erreurs, Eclipse le signale immédiatement. Si l'erreur est sémantique, il vous suggère des corrections possibles
- Lorsqu'on sauvegarde un fichier java, Eclipse le compile immédiatement.

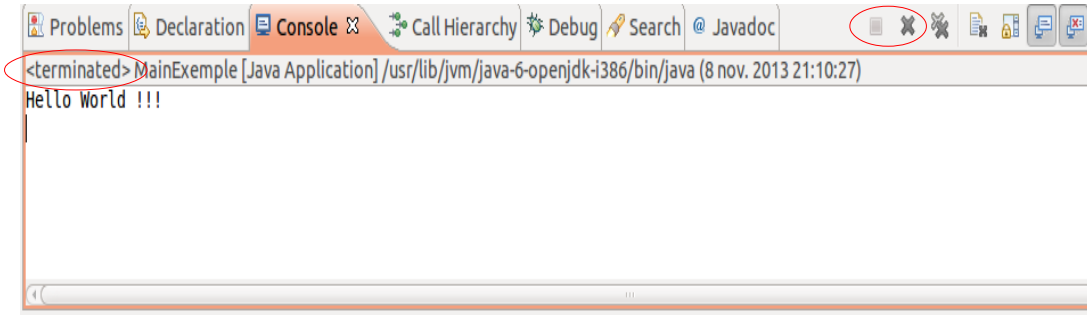
```java
public class MainExemple {

    /**
     *
     * @param args
     */
    public static void main(String[] args){
        // TODO Auto-generated méthode stub

        System.out.println ("Hello World !!!");
    }

}
```

Si on écrit le code précédent on exécute l'application par la commande . On aura l'affichage sur la console :



Dans la figure en haut de la console on indique le nom de la classe principale avec l'indication ***<terminated>*** (le programme est terminé) : Cette console est inactive.

En réalité, Eclipse crée un console pour chaque application lancée. Donc, on peut se trouver dans le cas où on aura cinq consoles. Le bouton permet de fermer la console inactive en cours. Et le bouton permet de fermer toutes les consoles

inactives.

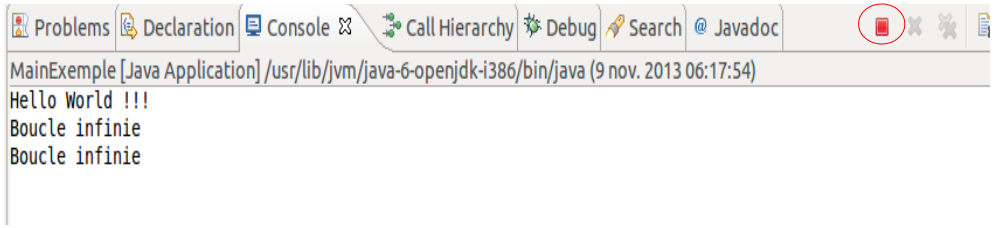
Le bouton  indique aussi que l'exécution du programme est terminée. Par contre, si on lance l'exemple suivant :

```
public class MainExemple {
    public static void main(String[] args) throws InterruptedException{
        System.out.println ("Hello World !!!");

        while (true){
            Thread.sleep(5000);
            System.out.println ("Boucle infinie");
        }
    }

}
```

On va avoir un affichage chaque cinq seconde: *boucle infinie*



Ce programme ne se termine pas à cause de la boule infinie *while (true)*. Pour l'arrêter, il suffit de clique sur le bouton  .

Le bouton  (à droite de la console) permet d'afficher la liste toutes les consoles (actives et inactives).

3. Débogage

Il arrive souvent qu'on aura des erreurs logiques dans nos programmes, de telle sorte de ne pas avoirs les résultats attendus à partir du programme. Ce type d'erreurs s'appellent : bugs.

Pour y remédier, Eclipse offre un un mécanisme de débogage offrant les possibilités suivantes :

✔ L'indication des points-d'arrêt (Breakpoint) : un breakpoint est une marque qu'on pose sur une ligne afin de causer l'arrêt (suspension) de l'exécution lorsque cette dernière arrive à cette ligne.

✔ Lors d'un arrêt, on peut examiner les valeurs des différents attributs

✔ À partir de ce point, on peut continuer l'exécution pas-à-pas.

La figure Fig.7. montre la perspective ***Debug***. Dans cette perspective, on trouve les vues :

– *Debug* : montrant l'ensemble de threads du programmes
– *(x)= Varibles* : pour les instances et les variables locales
– *Breakpoint* : affiche la liste des breakpoints

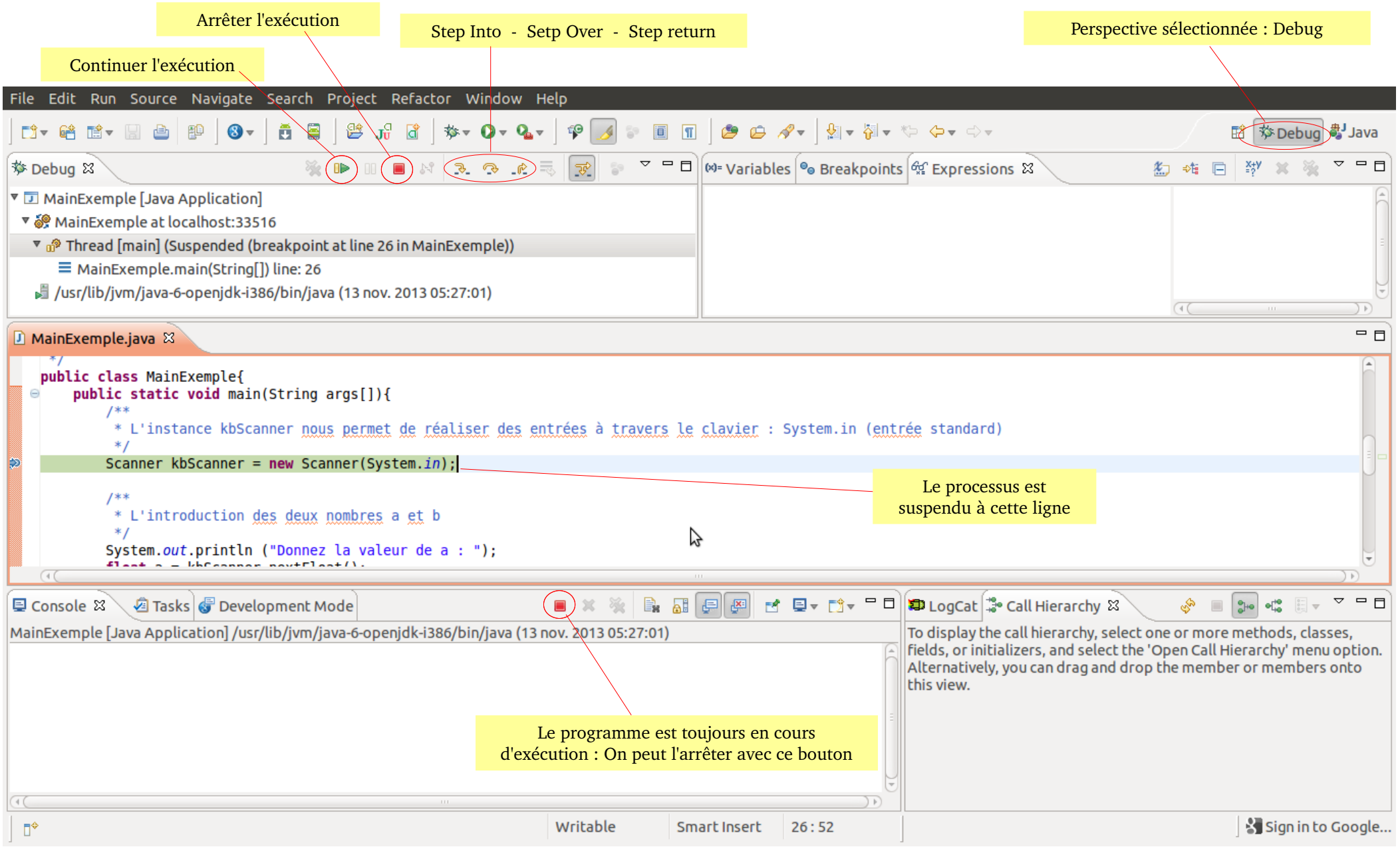


Fig7. Perspective Debug