	<i>Université de Corse - Pasquale PAOLI</i>	
	Diplôme : Licence informatique 3ème année	2020-2021
	UE : Ateliers de programmation	
	Environnement Eclipse-Java -Téléchargement et installation -Découverte de l'IDE Eclipse Enseignants : Paul-Antoine BISGAMBIGLIA, Marie-Laure NIVET, Evelyne VITTORI	

ECLIPSE se définit comme une *plateforme de développement universelle* destinée à supporter toutes sortes de langages. Sa version de base est dédiée au langage de programmation Java. Il propose en effet un environnement de développement intégré (EDI ou IDE pour Integrated Development Environment) constituant un véritable assistant pour le programmeur Java : le JDT (*Java Development Tooling*).

Les environnements associés à d'autres langages de programmation tels que C, C++, Python (et même COBOL !!) peuvent ensuite être « ajoutés » à la version de base sous la forme de briques logicielles (« plug-ins »).

Eclipse est ainsi un projet à architecture ouverte (« open source ») et extensible. A l'origine financé par IBM à travers sa filiale OTI (Object Technologies International), ECLIPSE est aujourd'hui géré par un consortium de développement « eclipse.org ».

Il est téléchargeable gratuitement sur le site du consortium <http://www.eclipse.org>.

Dans le cadre des séances de travaux pratiques, notre objectif n'est pas d'utiliser de manière exhaustive tous les aspects de l'outil Eclipse. Nous nous limiterons donc à la présentation des fonctionnalités minimum afin de nous concentrer sur l'essentiel de notre travail à savoir l'apprentissage des concepts orientés objet en Java.

Partie I – Téléchargement et installation de l'environnement de développement

I. Installation du JDK

1. Téléchargez la dernière version du JDK sur le site d'Oracle

✚ Adresse du site :

<https://www.oracle.com/java/technologies/javase-downloads.html>

✚ Dans la section Java SE Development Kit 14.0.2:

✚ Dans la colonne Download, sélectionnez la version correspondant à votre système (Windowsx64 par exemple) et cochez la case d'acceptation de la licence

✚ Choisissez d'enregistrer le fichier à télécharger

✚ Installez ensuite le logiciel en lançant l'exécution du fichier téléchargé et en suivant les instructions.

2. Retournez sur le site d'Oracle et téléchargez la documentation

<https://www.oracle.com/java/technologies/javase-jdk14-doc-downloads.html>

3. Installez la documentation en décompressant le fichier ZIP téléchargé dans le répertoire du JDK (création de la branche \docs\...) vous pourrez ainsi avoir accès à la documentation même lorsque vous n'êtes pas connecté à internet.

Remarque

Sous MacOS, le jdk. s'installe dans le dossier /Bibliothèques/Java/JavaVirtualMachines

II. Installation d'Eclipse

1. Téléchargez la dernière version de l'environnement Eclipse sur le site d'eclipse (Eclipse 2020-06)
 - + Adresse du site : <http://www.eclipse.org/downloads/>
 - + Choisissez un site de téléchargement en Europe
2. Pour installer Eclipse, il suffit d'exécuter le fichier télécharger (s'il s'agit d'un fichier d'installation (.dmg).ou de le décompresser s'il s'agit d'un fichier zip (une branche \eclipse\ ... sera créée avec l'ensemble des fichiers nécessaires) ou de l'exécuter s'il s'agit d'un fichier d'installation (.dmg).

Choisissez l'option Eclipse IDE for java developers.

Attention : Ne créez pas de répertoire « eclipse », celui-ci sera créé automatiquement dans le répertoire que vous aurez choisi (exemple D:\TPJava)

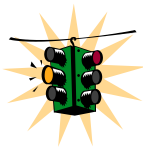
Partie II – Initiation à l'environnement de développement ECLIPSE

1. Workspace (Espace de travail)

Lors du premier lancement d'Eclipse, vous devrez définir votre espace de travail (workspace) c'est à dire le répertoire dans le quel se trouveront vos fichiers.

Par la suite, vous pourrez le modifier en utilisant l'option "switch workspace".

Attention au risque de perte de fichiers



Lors des séances prochaines, soyez bien attentifs à toujours vous situer dans le bon espace de travail en utilisant l'option Switch Workspace du menu File. Si vous constatez que vous êtes bien dans le workspace souhaité, vous pourrez cliquer sur le bouton Cancel.

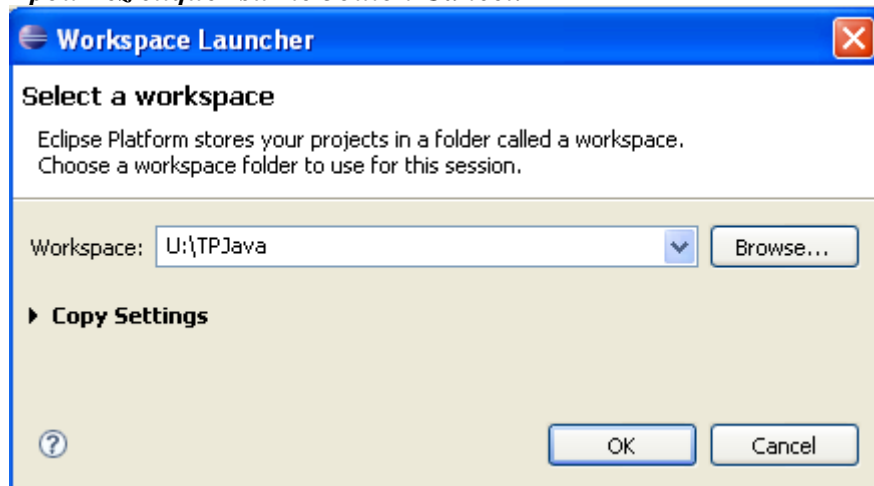
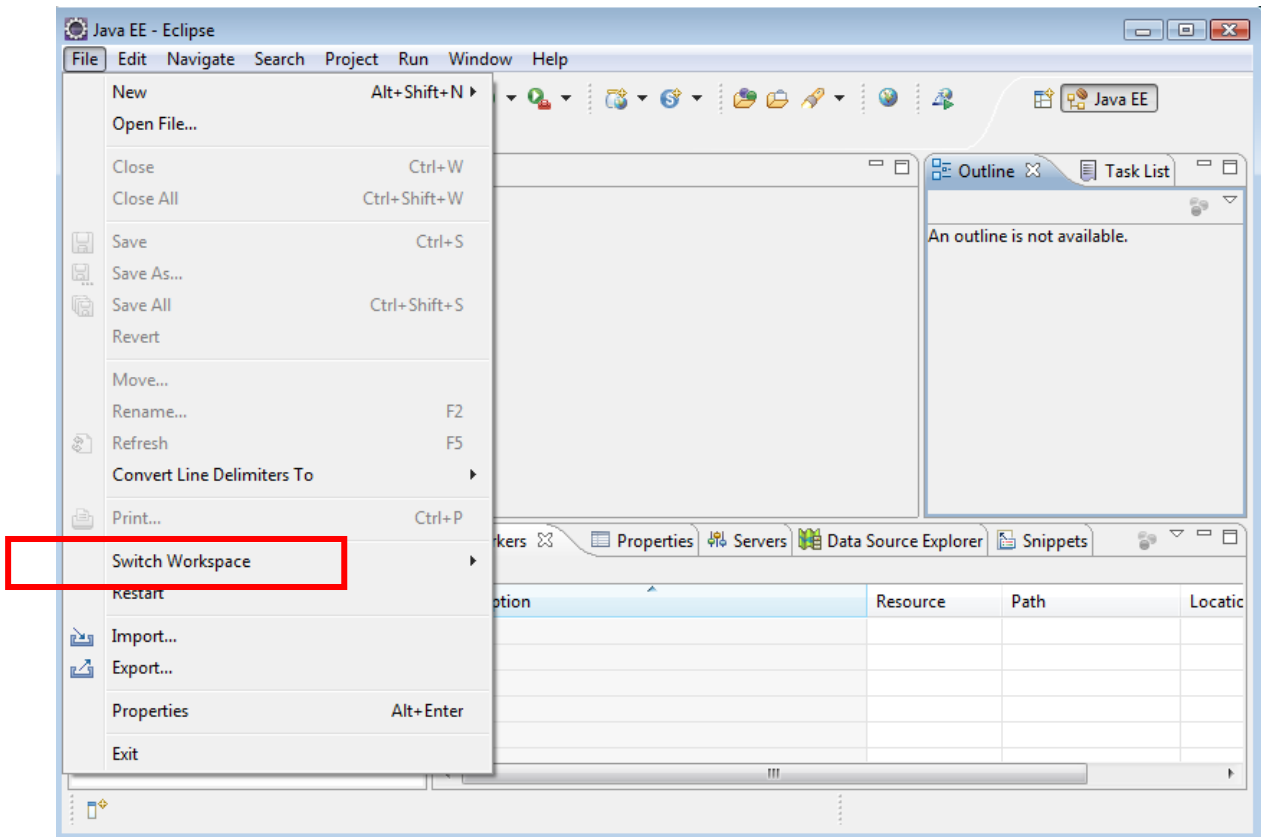


Figure 1 -Création d'un Workspace






Choisissez l'option *Switch Workspace* du menu *File* et prenez soin de définir la localisation de votre « workspace » sur votre espace de stockage personnel (U:\TpJava par exemple cf. Figure 1) .

Vous pouvez vérifier à l'aide du gestionnaire de fichiers Windows qu'un nouveau répertoire portant le nom que vous avez attribué à votre workspace (TpJava dans notre exemple) a été créé à l'emplacement spécifié (ici sur U:).



2. Lancement d'Eclipse

L'écran de lancement d'Eclipse vous propose plusieurs options de découverte (en anglais):

	<i>Tutoriaux</i>	Didacticiels de découverte
	<i>Samples</i>	découverte à partir d'exemples téléchargés
	<i>What's new</i>	Nouveautés de la version considérée



Vous pourrez tester ces différentes options plus tard, pour l'instant vous allez accéder au véritable environnement Eclipse (Surface de travail ou **Workbench**) en cliquant sur l'icône



située en haut à droite de l'écran d'accueil.

Remarque : La page d'accueil vous propose également différents boutons permettant d'accéder à des pages de documentation relatives à différentes spécialités de développement que nous n'allons pas étudier cette année (webServices, XML Tools, JavaEE, ect...).

3. Perspective Java

Vous vous trouvez à présent dans la perspective Java. Cet espace de travail est composé d'un ensemble de sous-fenêtres qui représentent soit des vues (**Views**) soit des éditeurs (**Editors**). Plusieurs vues et zones d'édition peuvent être empilées dans une sous-fenêtre; dans ce cas un onglet permet de sélectionner la vue ou la zone d'édition qui doit être affichée.

Le choix et la disposition des sous-fenêtres forment un arrangement appelé **Perspective**. Un certain nombre de perspectives sont pré-définies (*Java*, *Debug*, ...) et l'utilisateur peut rapidement commuter entre ces perspectives selon la tâche à effectuer (à l'aide des boutons situés dans l'angle supérieur droit du *Workbench*).

Lors de la création de votre espace de travail, vous vous trouvez dans la perspective par défaut d'Eclipse IDE for JavaDeveloper, il s'agit de la perspective standard Java.

Si vous le souhaitez, vous pourrez par la suite définir vos propres perspectives, les nommer et les enregistrer (cf. menu Windows, → *Customize Perspectives* → *Perspective As...*).

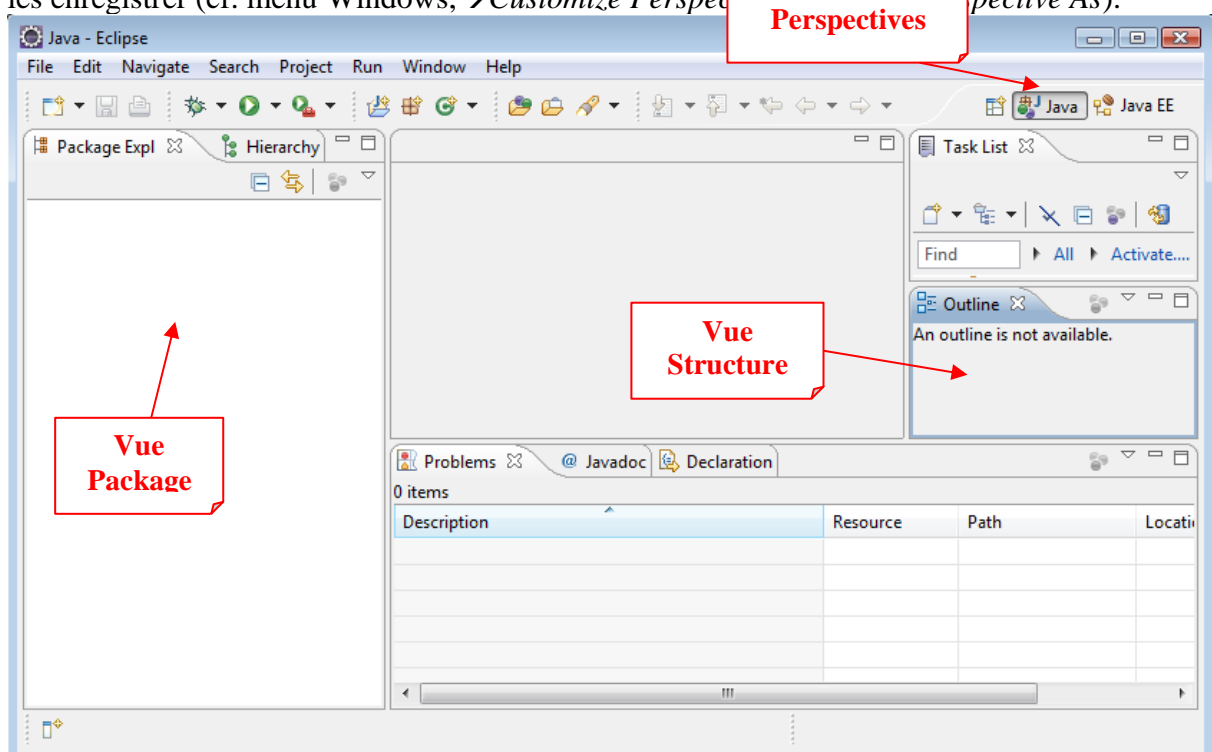


Figure 2 : Perspective « Java »


4. Création d'un projet

Pour développer un programme Java, il faut d'abord créer un **projet**. Plusieurs projets indépendants peuvent être créés dans le même Workspace. Tout au long des séances de TP, vous créerez ainsi un projet par sujet de TP et les placerez tous dans votre Workspace personnel (toujours le même évidemment !!).

Un projet peut contenir un nombre quelconque de ressources (classes, packages, dossiers, fichiers, applications, ...) et un certain nombre de propriétés lui sont associées (menu **Project** -

Properties).





Créez un nouveau projet en cliquant sur le bouton  « *New Java Project* » ou en sélectionnant le menu **File** puis **New** → **Project** → **Dossier Java** → **Java Project**.

Sélectionnez ensuite le bouton « *Next >* » pour passer à la fenêtre suivante. Dans la fenêtre qui s'affiche, saisissez TP1 dans la zone *project name* et choisissez de séparer dans deux répertoires distincts vos fichiers sources (fichiers .java) et vos fichiers « Bytecode » (fichiers .class) en cliquant sur l'option « *create separate source and output folders* » :

La fenêtre suivante de l'assistant permet de définir certaines propriétés du projet (ressources nécessaires, librairies, JRE, etc.). Pour les projets simples, il n'est pas nécessaire de modifier les valeurs proposées, vous pouvez donc cliquer directement sur le bouton « *Finish* » sans passer par « *Next* ».


Un nouveau répertoire portant le nom de votre projet (TP1) sera créé dans votre répertoire workspace. Ce répertoire TP1 comportera deux sous-répertoires :

-  **src** : répertoire des fichiers sources (.java).
-  **bin** : répertoire des fichiers (.class).

Dans la vue « *Package Explorer* », vous trouverez la liste des projets référencés dans le Workspace. Les projets peuvent être ouverts ou fermés à l'aide du menu **Project** (**Open Project** / **Close Project**). Il est conseillé de fermer les projets sur lesquels vous ne travaillez pas car les projets ouverts consomment des ressources et sont pris en considération lors de l'opération de compilation (rebuild).

5. Création d'une classe



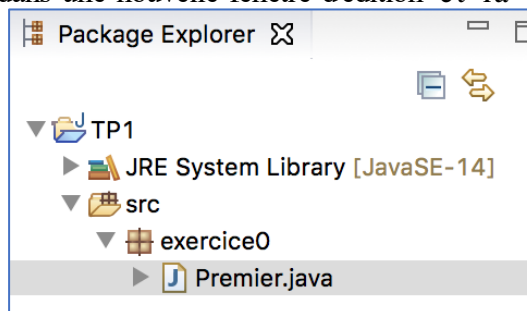
Définissez une nouvelle classe Java en cliquant sur le bouton  « *New Java Class* » ou en sélectionnant le menu **File** puis **New** → **Class**.

Dans la fenêtre qui s'affiche, définissez le nom de la classe « *Premier* » dans le champ *Name* et sélectionnez la case à cocher « *public static void main...* » afin de créer une classe comportant une méthode *main*.

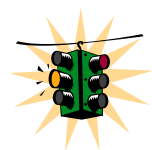
Définissez le nom du package « *exercice0* » dans le champ *Package* afin que la classe soit placée dans le nouveau package ainsi créé ***exercice0***.



Cliquez sur le bouton *Finish* afin de terminer la création de la classe. Le fichier source doit alors s'ouvrir dans une nouvelle fenêtre d'édition et la vue « *Package Explorer* » doit se présenter ainsi :



Explorer » doit se présenter ainsi :



Attention : Dans *Eclipse*, le nom du package détermine non seulement l'emplacement du fichier compilé (comme l'exige le langage Java) mais également l'emplacement du fichier source (sous-répertoire dans l'arborescence du projet).

Si l'on laisse le champ *Package* vide, la classe sera créée dans le paquetage par défaut. Le fichier source sera placé directement dans le répertoire *src* du projet.

Si l'on précise un nom de package (ici : `exercice0`), Eclipse va créer un sous-répertoire du répertoire `src` et y placer le fichier `.java`.

6. Edition d'un fichier source



S'il n'est pas déjà ouvert, ouvrez le fichier source de la classe `Premier` dans l'éditeur en double-cliquant sur son nom dans la vue *Package Explorer*.



Familiarisez vous avec les différentes fonctionnalités de l'éditeur en saisissant le code suivant ::


```
package exercice0
public class Premier {
    public static void main(String[] args) {
        int x=0, y, z=4;
        y = (2 * z) - 1 ;
        do {
            x += y;
            y -= 2;
        } while ((y >= 1) && (x != -1)) ;
        System.out.println("Résultat :" + x);
    }
}
```

Remarques

L'éditeur d'Eclipse possède beaucoup de fonctionnalités permettant de simplifier l'écriture de code *Java* et de nombreux aspects sont personnalisables (menu `Window - Preferences - Java - Editor`).

- + L'utilisation simultanée des touches **Ctrl-espace** procure une assistance directe permettant de compléter les instructions en cours d'écriture.
- + De nombreuses erreurs sont détectées instantanément durant l'édition du code source.
- + Certaines structures du code (classes, méthodes, ...) peuvent être temporairement masquées en cliquant sur les flèches placées dans la marge gauche de la fenêtre d'édition. Cette technique appelée *code folding* permet d'avoir une vue d'ensemble de la structure du code en masquant les détails de certaines régions. Un nouveau clic sur la flèche d'une structure réduite permet de revenir à l'affichage intégral du code.
- + La vue *Outline* présente de manière arborescente la structure du code en listant les noms des champs, des méthodes, etc. du fichier ouvert dans l'éditeur. En cliquant sur un élément de la vue *Outline*, l'éditeur se positionne sur la portion de code qui correspond à l'élément sélectionné.

7. Compilation

Dans la configuration par défaut, la compilation (*Incremental Build*) s'effectue lorsqu'on sauvegarde un fichier source (par le menu `File - Save` ou en cliquant sur le bouton  ou à l'aide du raccourci `Ctrl+S`).


Les erreurs de compilation sont signalées dans les marges gauche et droite de la fenêtre d'édition ainsi que dans la vue *Problems* où se trouvent les messages associés à chacune des erreurs.

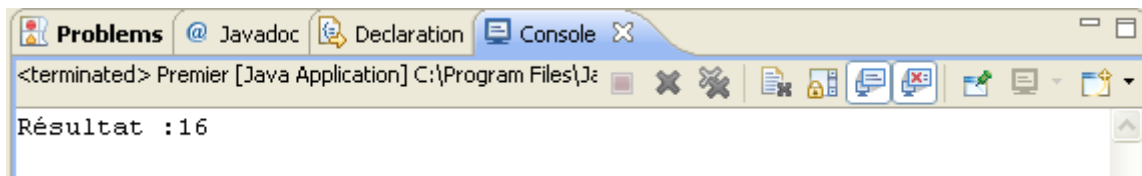


Compilez votre classe `Premier` en introduisant volontairement des erreurs de syntaxe afin de vous familiariser avec le compilateur.

8. Exécution



Exécutez à présent votre classe `Premier` en cliquant sur l'icône  « run Premier ». Vous devez obtenir l'affichage suivant dans la vue *Console* :




Par la suite, vous pourrez utiliser directement le bouton Run (ou les options du menu Run) pour compiler et exécuter. La compilation sera relancée automatiquement si elle s'avère nécessaire (modifications du code).

Remarques

L'exécution d'une application peut s'effectuer de différentes manières. Le menu *Run* rassemble la plupart des fonctions liées à l'exécution des applications.

Exécution directe


Pour une **application simple** (sans paramètres de lancement) on peut utiliser le

menu déroulant du bouton  et sélectionner l'option **Run As ► Java Application**. Cela créera automatiquement une configuration d'exécution par défaut.

Une application peut également être lancée en invoquant le menu contextuel du Package Explorer ou de la vue Outline en se positionnant préalablement sur la classe concernée. (bouton droit puis **Run ► Java Application**).

Il est également possible de définir manuellement une configuration d'exécution réutilisable à laquelle on donnera un nom et qui mémorisera toutes les informations de lancement.

Définition d'une configuration d'exécution

Pour définir une configuration d'exécution, on peut utiliser le menu déroulant du bouton  ► puis sélectionner l'option « **Run Configurations** »... qui démarre un assistant permettant de définir tous les paramètres de lancement (cf. Figure 3) et de les mémoriser sous un nom donné (champ "Name:").

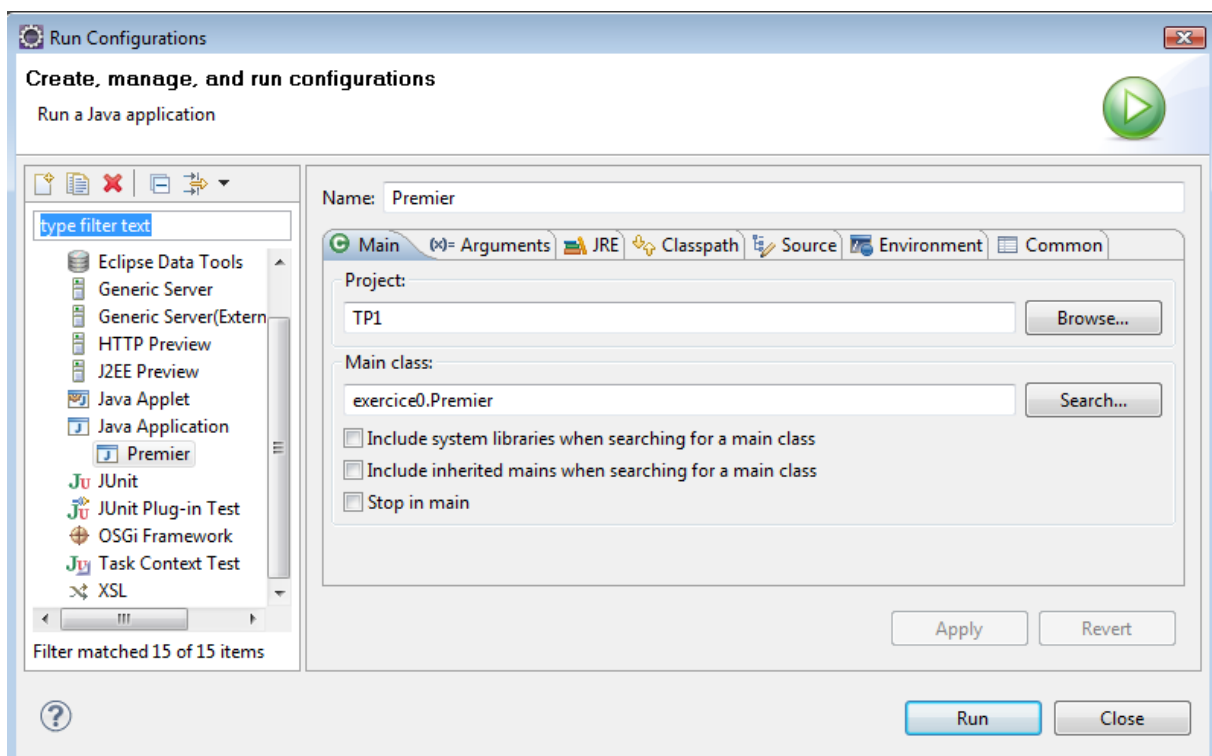



Figure 3 : Configuration d'exécution

Le lancement d'une application à partir d'une configuration d'exécution existante s'effectue en utilisant la même option Run . qui affichera la liste de toutes les configurations d'exécution disponibles. Une liste des dernières applications lancées est également affichée dans la partie haute du menu déroulant .




A l'aide de l'explorateur *Windows*, allez voir le contenu de votre répertoire *workspace* et identifiez la localisation de vos fichiers .java et .class.

9. Deboggage

Un débogueur (outil de recherche d'erreurs ou « bugs ») a pour objectif d'aider le développeur à localiser les erreurs d'exécution de ses programmes. Il permet également de mieux comprendre le fonctionnement d'un programme en l'exécutant instruction par instruction tout en visualisant l'évolution des variables.

La mise en œuvre du débogueur Eclipse consiste à lancer l'exécution d'une classe dans un mode particulier appelé « mode debug ».



Afin de vous familiariser avec ce débogueur, lancez à présent l'exécution de votre classe Premier en mode debug en cliquant sur l'icône  puis ▼ et sélectionnez « **Debug Configurations** » dans le menu déroulant. Dans la fenêtre de configuration qui s'affiche, cochez la case à cocher "Stop in main" afin de passer en mode d'exécution pas à pas (instruction par instruction). Cliquez sur le bouton *Debug* situé en bas de la fenêtre. Validez votre choix en appuyant sur le bouton *Yes*. Confirmez ensuite votre choix de changement de perspective dans la boîte de dialogue qui s'affiche.

Vous provoquez ainsi l'affichage d'une perspective pré-définie appelée Debug (cf. figure 4) et le lancement de l'exécution.

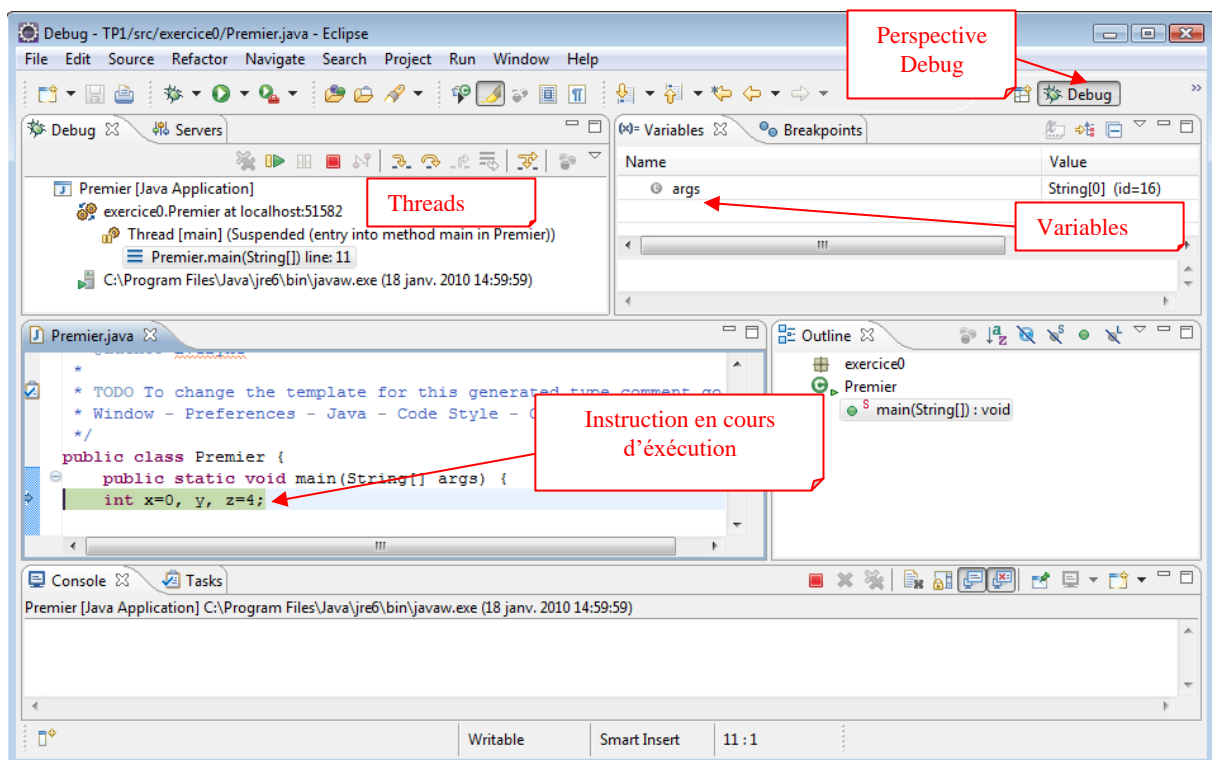


Figure 4 : Perspective Debug


Lorsque le programme est exécuté selon ce mode, le contenu des variables peut être consulté en utilisant la vue "Variables" ou en positionnant le curseur (dans le code source) sur la variable à


inspecter ce qui va provoquer l'affichage d'une bulle d'aide (*Tooltip*) indiquant le contenu de la variable.



Exécutez votre programme pas à pas à l'aide des commandes :

Step Over  / F6 : exécute l'instruction courante et place le curseur sur la ligne suivante,

Step Into  / F5 : exécute l'instruction courante et s'il s'agit d'une innovation de méthode, place le curseur sur l'instruction suivante à exécuter dans la méthode appelée


Resume ( / F8) : poursuit l'exécution jusqu'au prochain *Breakpoint*



: Arrêt de l'application.





Modifiez la valeur d'initialisation de la variable z. Essayez successivement les valeurs 5,6 et 7. Que calcule ce programme dans la variable x ?

Pour définir un point d'arrêt (*Breakpoint*) il suffit d'effectuer un double-clic dans la marge gauche de l'éditeur. Un point bleu () confirme visuellement l'emplacement du point d'arrêt (*attention : des points d'arrêt ne sont possibles qu'avant une instruction exécutable, on ne peut pas, par exemple, placer un point d'arrêt sur une déclaration de variable locale ou sur une accolade de début ou de fin de bloc d'instructions*).

10. Guides d'utilisation Eclipse

Pour aller plus loin et découvrir l'ensemble des fonctionnalités d'Eclipse, plusieurs outils sont à votre disposition :

-  Dans l'aide d'Eclipse (en anglais) : Choisissez l'option *Help Contents* du menu *Help*. Vous pouvez ensuite parcourir les tutoriaux proposés dans les sections *Getting Started* puis *Basic Tutorial* des deux rubriques suivantes :
 - ***Workbench User Guide*** : Découverte de l'environnement
 - ***Java Development User Guide*** : Apprentissage du JDT (Java Development Tooling) l'environnement de développement Java dans Eclipse.
-  Sur le Web, vous pouvez utiliser notamment le tutoriel très clair (en français !) de Jean Michel Doudoux disponible sur <http://jmdoudoux.developpez.com/> .