Programmation Orientée Objet (Java I)
*****

TD Introduction à l'environnement d'eclipse

Eclipse est un IDE, *Integrated Development Environment* (EDI environnement de développement intégré en français), c'est-à-dire un logiciel qui simplifie la programmation en proposant un certain nombre de raccourcis et d'aide à la programmation. Il est développé par IBM, est gratuit et disponible pour la plupart des systèmes d'exploitation.

Au fur et à mesure que vous programmez, eclipse compile automatiquement le code que vous écrivez, en soulignant en rouge ou jaune les problèmes qu'il décèle :

- Il souligne en rouge les parties du programme qui ne compilent pas,
- et en jaune les parties qui compilent mais peuvent éventuellement poser problème (on dit qu'eclipse lève un avertissement, ou *warning* en anglais).

Installation
Eclipse est simplement une interface, écrite en Java, qui va vous permettre de développer des programmes en Java. Ce logiciel peut donc être simplement vu comme un « enrobage » de Java. Eclipse peut également être utilisé pour le développement d'applications écrites avec d'autres langages de programmation.

La première chose à installer est donc le kit de développement de java (JDK) que vous trouverez à l'adresse : www.oracle.com/. Dans le menu cliquer sur products allez dans Java, cliquez sur **download**. Télécharger la **Java Platform, Standard Edition 17** par exemple. Vous devez alors accepter les conditions en cochant la case « Accept Licence Agreement » puis choisir la version de Java correspondant à votre système d'exploitation.
Ou allez directement sur ce lien : https://www.oracle.com/java/technologies/downloads/archive/
Et choisir la plateforme Java SE et choisir une version adaptée à votre système.

Qu'est-ce qu'un JDK ?
Le kit de développement Java, ou JDK, permet de compiler et d'exécuter le code Java sur votre ordinateur.
Sachez simplement que ce JDK contient :

*un compilateur permettant de traduire le code source Java en un code binaire optimisé pour n'importe quel système Windows, Linux, Mac ;

*une JVM, ou machine virtuelle Java. Il s'agit d'un logiciel sachant lire ce code binaire et l'exécuter sur le système de votre ordinateur.

L'environnement Java étant maintenant installé sur votre machine, eclipse peut alors être récupéré. Rendez-vous sur le site : www.eclipse.org/ide/ et récupérez le fichier Eclipse IDE for Java Developers qui correspond à votre système d'exploitation. Installez-le.

Utilisation
Avec Eclipse vous travaillez toujours au sein d'un **projet**.
Un projet est un ensemble de fichiers source JAVA et des informations associées (classpath, comment construire et exécuter le projet, librairies, etc...) correspondant à un développement donné.

L'IDE Eclipse stocke l'information associée à un projet dans un dossier projet (project folder) qui, en plus de vos fichiers sources et vos fichiers compilés, inclus toutes les ressources nécessaires à votre projet de développement ainsi que deux fichiers de configuration (.classpath et .project) utilisés par l'IDE et que vous ne devez bien entendu pas supprimer.

Tous vos projets sont regroupés dans un répertoire, le WorkSpace Eclipse, que vous devrez choisir lors du lancement d'Eclipse.

Lancement et « workspace »

Lors du premier lancement d'Eclipse, l'environnement vous demande de spécifier un « workspace » (voir Figure 1). Les workspaces (espaces de travail) d'Eclipse sont des répertoires dans lesquels sont regroupés les projets de développement.

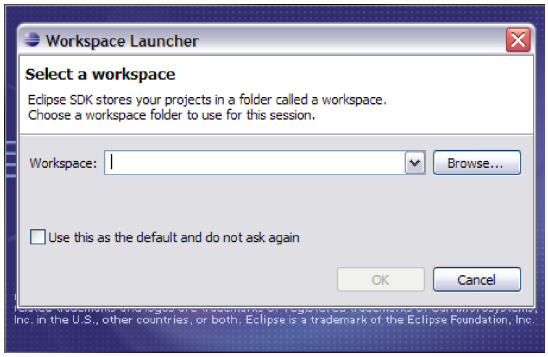


Figure 1 - Dialogue de sélection et de création d'un workspace.

Pour les enseignements de Java, nous allons créer un workspace spécifique lors du premier démarrage d'Eclipse.

Le bouton « Browse… » vous permet d'ouvrir un dialogue de parcours/sélection de fichier dans lequel vous allez pouvoir créer un répertoire.

A l'aide du dialogue de sélection de répertoires, créez le répertoire« MesProgrammesJava» (par exemple) sur votre répertoire personnel.

Lorsque vous validez, vous revenez à la fenêtre de choix de workspace qui contient dans la liste le répertoire nouvellement créé. Vous pouvez alors lancer l'environnement avec le bouton « OK ».

Changer de workspace

Il est possible de changer de workspace à tout moment. Le workspace en cours est alors fermé (ainsi que tous les projets qu'il contient). Pour cela, il suffit d'aller dans le menu « File » et de sélectionner « Switch workspace… ». La fenêtre de sélection de workspace apparaît alors afin de choisir un workspace existant ou d'en créer un nouveau comme vu précédemment.

Projets

Au sein des workspaces, Eclipse regroupe le code en **projets**. Un projet peut regrouper le code d'une application, d'une librairie, d'un module d'une application, etc. Pour les TPs de java, nous ferons dans la plupart des cas un projet par TP. Pour certains TP, vous aurez besoin de vos projets déjà crées. C'est pourquoi nous allons voir la création, l'importation et l'exportation de projets.

Création d'un projet

Nous allons créer un nouveau projet pour nos manipulations.

1. On peut accéder au guide de création de projets par deux méthodes : par le menu « File/New/Project… » ou par le menu contextuel du panneau « Package Explorer » (voir Figure 2).
2. Dans la première case, donnez un nom à votre projet, par exemple ProjetTest.
3. Normalement, le bon JRE est sélectionné. La version de Java à utiliser pour compiler et exécuter le projet (« JRE »), ainsi que des options avancées de compatibilité entre les versions de Java.
4. Cochez la case "create separate folders for source and class files" qui vous permet d'avoir un répertoire src pour vos sources et bin pour les fichiers compilés.
5. Cliquez sur Finish.
6. Votre projet est crée.

Nous allons observer ce qu'Eclipse a crée sur le disque pour que vous puissiez l'utiliser.

1. sur votre projet, cliquez-droit, et sélectionnez "properties" dans le menu contextuel. Regardez où se trouve votre projet sur l'ordinateur.
2. (en dehors d'eclipse) Ouvrez le dossier correspondant
3. Le répertoire bin contiendra tous les fichiers compilés
4. Le répertoire src contient toutes vos sources

5. Les autre répertoires, dont les noms commencent par un "." sont des répertoires utilisés par eclipse pour gérer le projet.

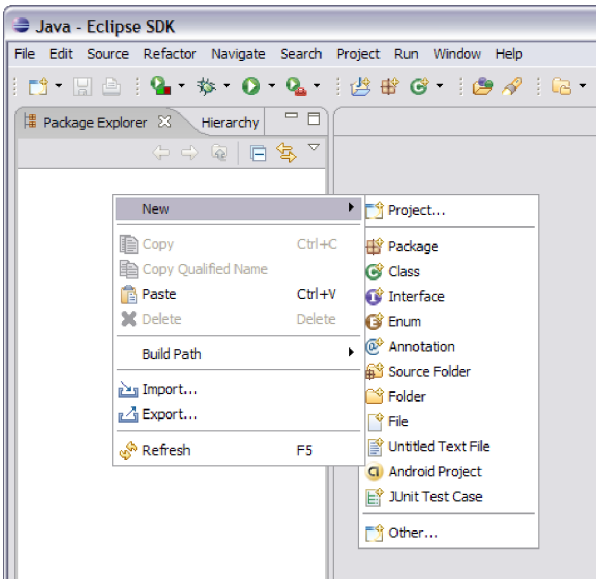


Figure 2 - Création d'un projet. Menu contextuel du panneau "Package Explorer".

Ajouter des fichiers au projet

L'ajout de nouveaux fichiers au projet (interfaces, classes Java, etc.) se fait par le menu « File/New… » lorsqu'un projet est sélectionné ou par clic droit sur un élément du projet auquel vous souhaitez ajouter un fichier.
Nous allons ajouter une classe au projet « ProjetTest ». Après un clic droit sur le projet et après avoir choisi « New/Class », la fenêtre de création de classe apparaît (Figure 3). Il faut spécifier le nom de la classe, et éventuellement : le « package » dans lequel elle se trouve, les interfaces qu'elle implante, les classes dont elle hérite, sa visibilité, etc.
Nous ne spécifieront ici que le nom de la classe : « Bonjour ». Une fois le nom saisi, le bouton « Finish » est activé et lorsque l'on clique dessus, le fichier « Bonjour.java » est créé et ajouté au projet. Il est automatiquement ouvert dans l'éditeur de code et, selon ce qu'il a été spécifié comme options dans la fenêtre de création, certaines parties de code sont déjà écrites (Figure 3).

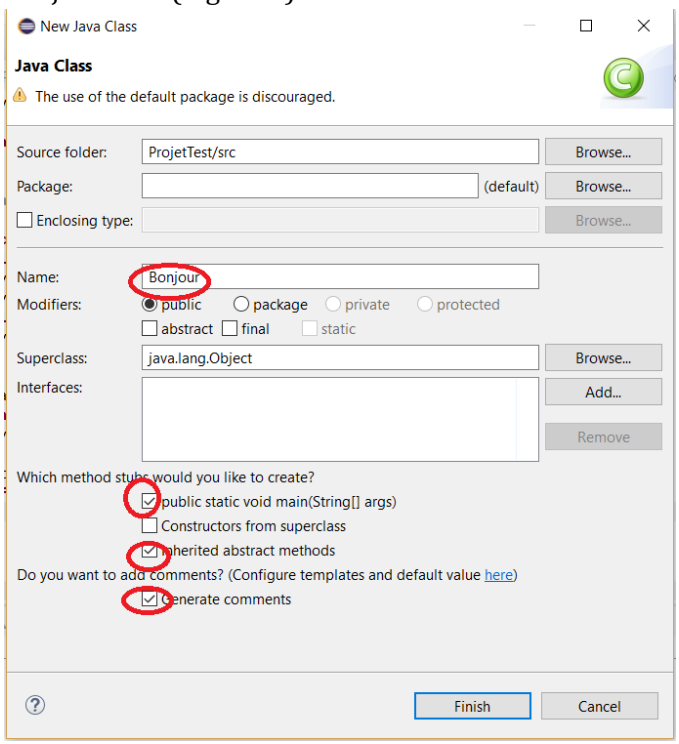


Figure 3 - Création d'une nouvelle classe

1. indiquez le nom de la classe à créer. Ici Bonjour.
2. demandez à l'assistant de générer un squelette de classe contenant une méthode main en cochant la case
3. demandez à l'assistant de générer les commentaires (que vous pouvez personnaliser en configurant les propriétés de votre projet).
4. en cliquant sur le bouton Finish, vous provoquez la création du fichier Bonjour.java dans les sources de votre projet et l'ouverture de l'éditeur sur celui-ci.

Ecriture du fichier source

Exercice 1

Il suffit ensuite de recopier les lignes qui suivent et d'enregistrer le fichier obtenu.

```
/* Premier programme java : afficher bonjour.  */
public class Bonjour {
 public static void main(String args[]) {
  System.out.println("Bonjour !");
 }      }
```

Quelques explications :

1. Les parties de code situées entre **/*** et ***/** sont des commentaires qui ne seront pas compilés.
2. La première instruction **public class Bonjour** déclare que nous créons une classe publique (accessible par tous) qui se nomme Bonjour. Cette classe doit être enregistrée dans un fichier **de même nom** et d'extension .java. Attention, java différencie les majuscules et les minuscules. La classe bonjour est différente de la classe Bonjour, et le fichier bonjour.java n'est pas l'équivalent du fichier Bonjour.java.
3. Java étant un langage orienté objets, tout fichier source définit une classe, c'est à dire un objet possédant des champs et des méthodes. La définition de la classe se fait dans un bloc d'instructions contenu entre 2 accolades.
4. Notre classe bonjour contient une unique méthode annoncée par l'instruction **public static void main(String args[]).** Il s'agit de la méthode principale d'un programme, celle qui sera exécutée lors du lancement. Le nom et la déclaration de cette méthode ne peuvent pas être modifiés.
5. La définition d'une méthode est une suite d'instructions située entre 2 nouvelles accolades. Dans le cas qui nous intéresse, la méthode main n'a qu'une seule instruction qui provoque l'affichage du message "Bonjour!", par l'intermédiaire de la méthode **System.out.println**. Les lignes contenant une instruction simple comme celle-ci doivent se terminer par un **point-virgule**.

L'éditeur n'attend pas toujours qu'une compilation ait été lancée pour vous signaler d'éventuelles erreurs. Au fur et à mesure que vous tapez votre texte, la syntaxe java est vérifiée et en cas d'erreur une petite croix rouge est mis en face de la ligne correspondante. En positionnant le curseur de la souris sur cette croix vous aurez un message d'explication indiquant la nature de l'erreur.

Compilation, tests et exécution

Un des avantages d'Eclipse et de fournir un environnement intégré qui facilite la création, la compilation, les tests et l'exécution de projets java pouvant contenir de nombreux fichiers de code, repartis dans différents paquetages et référençant de nombreuses librairies.

Compilation

Par défaut, lorsqu'un projet est créé, Eclipse active l'option de « compilation automatique ».Ainsi, votre projet est compilé en temps réel, dès que vous tapez du code et sauvegardez un fichier ou que vous en lancez l'exécution. Eclipse fournit en plus de nombreux mécanismes pour présentez les erreurs de compilation.

Exécution

Pour exécuter une application Java, procédez comme suit :

1. cliquez bouton droit sur la classe contenant le main dans l'explorateur de packages.
2. dans le menu contextuel, sélectionnez l'item Run As

3. dans le sous menu sélectionnez l'item Java Application
L'affichage des sorties de votre exécution s'effectue dans la fenêtre Console située en bas de votre WorkBench (si cette fenêtre est absente, vous pouvez la faire apparaître avec le menu **Window -> ShowView -> Console**).
En haut de la console, le statut de l'exécution est affiché : <terminated> pour l'application Java Bonjour.

Exercice 2 : **réexécution d'une classe**

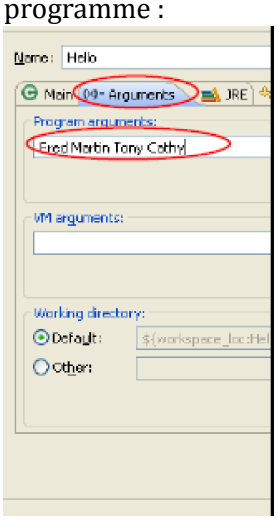
1. Que se passe t-il si l'on tape sysout et que l'on appuie sur Ctrl et espace dans un main ?
2. Même question en tapant main puis ctrl et espace dans une classe ?

Exercice 3 : **Exécution avec arguments sur la ligne de commandes**

Nous allons maintenant compléter le programme Bonjour afin qu'il puisse lire une suite de noms sur la ligne de commandes (paramètre args de Main) et pour chaque nom lu afficher une
chaîne Bonjour ...

```
public static void main(String[] args) {
for (String mot : args)
        System.out.println("Bonjour " + mot); }
```

Pour lancer ce programme avec des arguments depuis Eclipse il faut modifier la configuration du Run du programme :

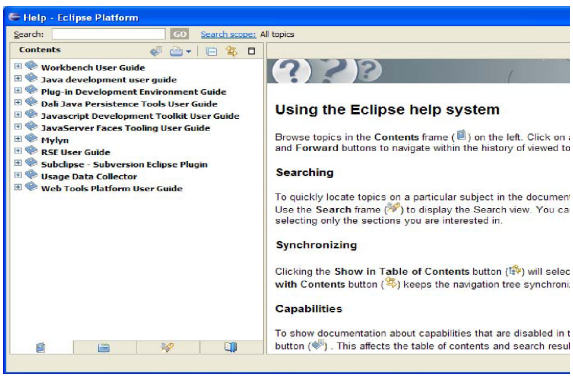


1. cliquez sur la flèche-bas à la droite du bouton Run.
2. Sélectionner l'item O**pen Run Dialog...** dans le menu contextuel
3. La fenêtre de configuration des exécutions est affichée
4. Dans cette fenêtre, sélectionnez l'application Java Bonjour, et dans celle-ci, l'onglet **Arguments**
5. Dans le champ texte **Program Arguments** écrivez les arguments que vous souhaitez transmettre à votre programme lors de son exécution (séparés par des espaces)
6. Sauver cette nouvelle configuration en cliquant sur **Apply** puis lancez l'exécution en cliquant sur le bouton **Run.** Vérifiez l'affichage de votre console ?

Exercice 4 :

Modifiez la classe Java Bonjour pour appeler dans main une méthode direBonjour, prenant en paramètre le nom de la personne et qui affiche bonjour nom à l'écran.

La documentation en ligne d'Eclipse
Pour profiter aux mieux des très nombreuses fonctionnalités de cet environnement, nous vous conseillons de parcourir la documentation en ligne. Pour obtenir cette aide, menu **Help-->Help Contents**. La Fenêtre d'aide apparaît.



Cette documentation est très vaste. Pour débuter concentrez-vous sur le Tutorial de base du Workbench User Guide ainsi que sur le tutorial de base du Java Development User Guide Il ne vous a été présenté ici qu'une infime partie des possibilités et fonctionnalités d'Eclipse, constituant la base dont vous aurez besoin pour démarrer sereinement les TP de Java. Nous vous encourageons bien évidemment à aller plus loin et explorer par vous-même les nombreuses possibilités que vous offre cet environnement.

Pour allez plus loin, ou en cas de « panne », vous pouvez consulter :

- Le site officiel : http://www.eclipse.org/
- Documentation JDK 17 :

https://docs.oracle.com/en/java/javase/17/docs/api/index.html

Eclipse : raccourcis utiles

```
ctrl + f11 => exécuter le code
ctrl + espace => auto-complétion
ctrl + maj + o => auto-import
alt + maj + s => menu actions rapides
D'autres raccourcis :
http://www.vogella.com/tutorials/EclipseShortcuts/article.html
```