iUT ORSAY
Département Informatique

DUT Informatique
Module JAVA Apprentis
2008 / 2009

Travaux Pratiques n° 1 : Eclipse

Nom(s) :
Groupe :
Date :

Objectifs : Apprendre à se servir du logiciel Eclipse pour le développement d'applications JAVA (créer de nouveaux projets, paquetages et classes, apprendre à exécuter les programmes et à générer une javadoc).

1 Présentation d'Eclipse

Eclipse est un IDE (Integrated Development Environment), c'est à dire un logiciel spécialisé dans l'aide au développement d'applications. Les IDE intègrent au minimum les trois outils indispensables au développement : un éditeur de texte, un compilateur et un débogueur. D'autres IDE très répendus sont Visual Studio de Microsoft, Windev de PC Soft ou encore Delphi de Borland. Eclipse, développé à l'origine par IBM, a la particularité d'être un logiciel libre écrit en JAVA, on le trouve donc gratuitement sur toutes les plateformes (Windows, Linux, MacOSX, Solaris, BSD...). Prévu au départ pour le développement d'applications JAVA, sa grande modularité fait qu'il peut facilement être utilisé pour d'autres langages grâce à des *plugins*, en particulier C/C++, PHP, HTML ou encore Python. C'est aujourd'hui un outil largement utilisé dans le monde industriel. Pour télécharger Eclipse et obtenir toutes les informations utiles :

`http://www.eclipse.org`

2 L'environnement Eclipse

L'environnement de travail sous Eclipse est divisé par défaut en quatre parties principales qu'il est possible de modifier selon ses préférences (ce que nous déconseillons car il est extrêmement facile de ne plus s'y retrouver) :

- La partie de gauche est un explorateur qui vous permettra de naviguer dans vos projets, paquetages, classes etc.
- La partie centrale est l'éditeur de texte. Vous pourrez éditer plusiers textes en même temps, ils seront alors séparés en onglets.
- La partie de gauche vous permet de retrouver facilement les divers éléments de votre programme (paquetages, classes, méthodes etc).
- La partie du bas vous présente des onglets correspondants aux sorties des divers outils (fenêtre d'exécution du programme, documentation du programme, débogueur).

L'image en Figure 1 vous donne une vue d'ensemble de la fenêtre Eclipse.

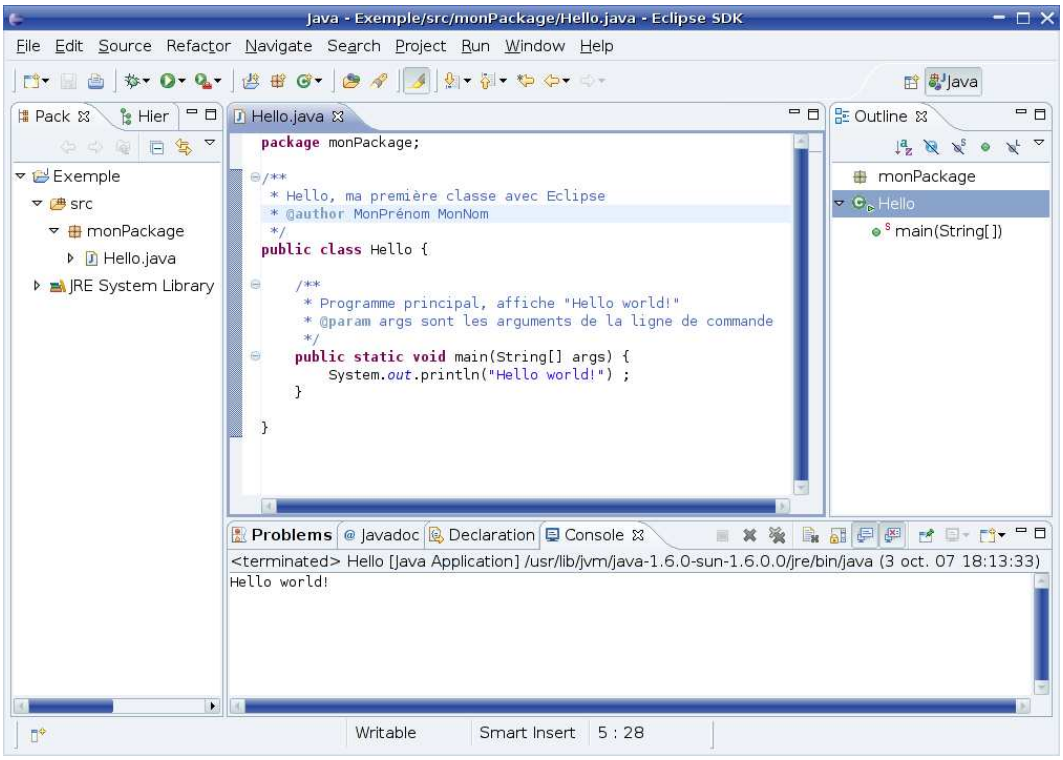

FIG. 1 – L'environnement Eclipse

3 Lancement d'Eclipse

Lors de son premier lancement, Eclipse vous demandera où se situe votre espace de travail, ou *workspace*. Il s'agît d'un répertoire où Eclipse dépose ses fichiers de configuration et vos fichiers de travail. Vous ne devriez jamais aller toucher aux fichiers contenus dans ce répertoire sans passer par l'interface d'Eclipse, sans quoi le logiciel fonctionnera mal. Veillez à choisir un répertoire dans votre espace personnel (sur `Z:\`) pour être certain de le retrouver sous Windows ou Linux, ou encore après une suppression de votre profil itinérant.

Démarrez Eclipse directement depuis l'icône sur votre bureau, ou dans le menu *Démarrer* ou s'il n'est pas présent, en allant le chercher directement dans `C:\Program Files\Eclipse`. Si c'est la première fois que vous démarrez Eclipse, fermez l'onglet *Welcome* qui s'affiche pour accéder à l'environnement de travail.

4 Un premier projet

Afin de pouvoir travailler simplement sur plusieurs applications distinctes en même temps, Eclipse utilise la notion de *projet*. La création de nouveaux projets JAVA se fait par la séquence *File→New→Project...*, en choisissant *Java Project* dans la boîte de dialogue, puis cliquer *Next*, donner un nom à son nouveau projet et enfin *Finish*.

Créez un nouveau projet de nom `Exemple`.

5 Un premier paquetage

Les applications JAVA sont découpées en *paquetages* ou *packages* qui permettent de classer les codes sources suivant leur utilité ou fonctionnalités. Ce point sera abordé plus tard en cours, mais dès maintenant nous allons les utiliser pour en prendre l'habitude. Dans un projet, on crée un nouveau paquetage en faisant un clic droit sur le nom du projet, puis *New→Package*, en donnant un nom au nouveau paquetage (en JAVA, la convention veut que les noms de paquetage commencent par une minuscule) et enfin *Finish*.

Créez un nouveau paquetage de nom `monPackage` dans le projet `Exemple`.

6 Une première classe

L'unité de base des codes source JAVA est la *classe*, un paquetage pouvant regrouper plusieurs classes. On crée une classe dans un paquetage donné sous Eclipse en faisant un clic droit sur le nom du paquetage, puis *New→Class*, en donnant un nom à la nouvelle classe (en JAVA, la convention veut que les noms de classe commencent par une majuscule) et enfin *Finish*. La fenêtre de création de la classe (montrée en Figure 2) permet de choisir différentes particularités pour générer automatiquement un code JAVA qui ne restera qu'à remplir. En particulier vous pouvez choisir d'ajouter la méthode `main` qui servira de point d'entrée à votre programme.

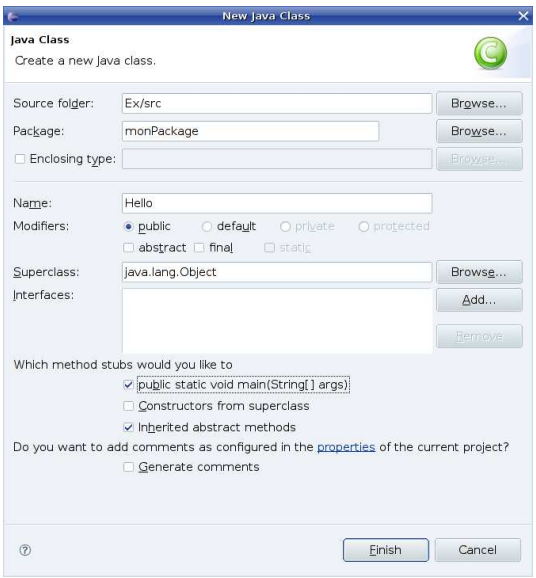

FIG. 2 – Boîte de création d'une classe

Créez une nouvelle classe `Hello` dans le paquetage `monPackage`, le code devra correspondre exactement, **y compris les commentaires**, à celui montré en Figure 3.

7 Une première exécution

Pour lancer l'exécution d'un programme en passant par la méthode `main` d'une classe, faire un clic droit sur le nom de la classe, puis *Run As→Java Application*. Le résultat de l'exécution s'affiche alors dans l'onglet *Console* de la partie basse de l'environnement.

```
package monPackage;

/**
 * Hello, ma première classe avec Eclipse
 * @author MonPrénom MonNom
 */
public class Hello {

        /**
         * Programme principal, affiche "Hello world!"
         * @param args sont les arguments de la ligne de commande
         */
        public static void main(String[] args) {
                System.out.println("Hello world!");
        }

}
```

FIG. 3 – Code du fichier *Hello.java*

Lancez l'exécution de votre programme. Que s'affiche-t-il dans l'onglet *Console* (si cet onglet n'est pas ouvert, l'ajouter par *Window→Show View→Console)* ?

Peut-on mettre plusieurs classes dans un fichier (testez) ? Si oui, dans quelles conditions ?

8 Arguments de la ligne de commande

Sous Eclipse, le programme est lancé depuis une interface graphique, pour donner des arguments de la ligne de commande, il sera nécessaire de les préciser dans *Run→Open Run Dialog*, de choisir la *Java Application* correctement et de remplir l'onglet *Arguments* puis *Program arguments* comme on le souhaite. Toutes les exécutions suivantes tiendront compte des arguments rentrés.

Modifiez légèrement votre programme pour afficher un argument de la ligne de commande.

9 Une première javadoc

JAVA dispose d'un système de documentation automatique particulièrement puissant : la javadoc. On peut générer automatiquement de la documentation pour un code en faisant la manipulation *Project→Generate Javadoc...* Il sera sans doute nécessaire de préciser où se trouve la commande javadoc. Pour cela après *Project→Generate Javadoc* faire *Configure* pour *Javadoc command* et aller désigner `C:\Program Files\Java\jdk`x.x.x`\bin\javadoc.exe`. Ensuite

Finish. La javadoc du programme sera alors visible dans la partie navigation (*doc→index.html*) et dans l'onglet *Javadoc* de la partie basse de l'environnement (par exemple cliquez sur le nom de la classe pour avoir ses informations dans cet onglet).

Générez la Javadoc de votre programme, comment se présente-t-elle ?

Quelle est l'utilité des marqueurs (ou *tags*) tels que `@author` ou `@param` ? Donnez (en cherchant sur internet par exemple) d'autres exemples de marqueurs et leur utilité.

Ouvrez un navigateur web et visitez la javadoc de JAVA à l'adresse `http://java.sun.com/javase/6/docs/api`. Que fait la méthode `parseInt` de la classe `Integer` ?

10 Exportation et importation des projets

Pour importer et exporter ses projets crées sous Eclipse pour les utiliser par exemple sur une autre machine, il faut éviter d'utiliser directement les fichiers dans votre *workspace*.

Pour **exporter** correctement un projet, faire un clic droit sur le nom du projet, puis *export*, choisir dans la boîte de dialogue *General→Archive File→Next*, choisir le chemin et le nom de l'archive dans laquelle enregistrer le projet (par *Browse*) et enfin *Finish*.

Pour **importer** correctement un projet, faire *File→import*, choisir dans la boîte de dialogue *General→Existing Projects into Workspace→Next*, choisir *Select archive file* et le chemin et le nom de l'archive (par *Browse*) et enfin *Finish*. Attention, cela ne fonctionnera pas si un projet du même nom que celui que vous tentez d'importer est déjà présent dans votre workspace.

Copiez votre projet *Exemple* en le renommant (pour cela clic droit sur le nom du projet puis *Copy*, clic droit dans la partie navigation puis *Paste* en donnant un nouveau nom), exportez votre copie, détruisez la copie (clic droit sur le nom du projet et *Delete* en choisissant *Also delete content*), puis importez la copie.