Introduction à Eclipse

Eclipse IDE est un environnement de développement intégré libre (le terme Eclipse désigne également le projet correspondant, lancé par IBM) extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation. Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions. [*Wikipedia*]

Dans les prochains TP de Java, nous utiliserons Eclipse comme environnement de développement. Cette petite introduction vous guidera dans vos premier pas avec Eclipse en vous en présentant les outils de base.

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. Ainsi, il est possible d'avoir plusieurs espaces de travail, dédiés à des langages de programmation différents, des buts différents, ou par exemple des enseignements différents.

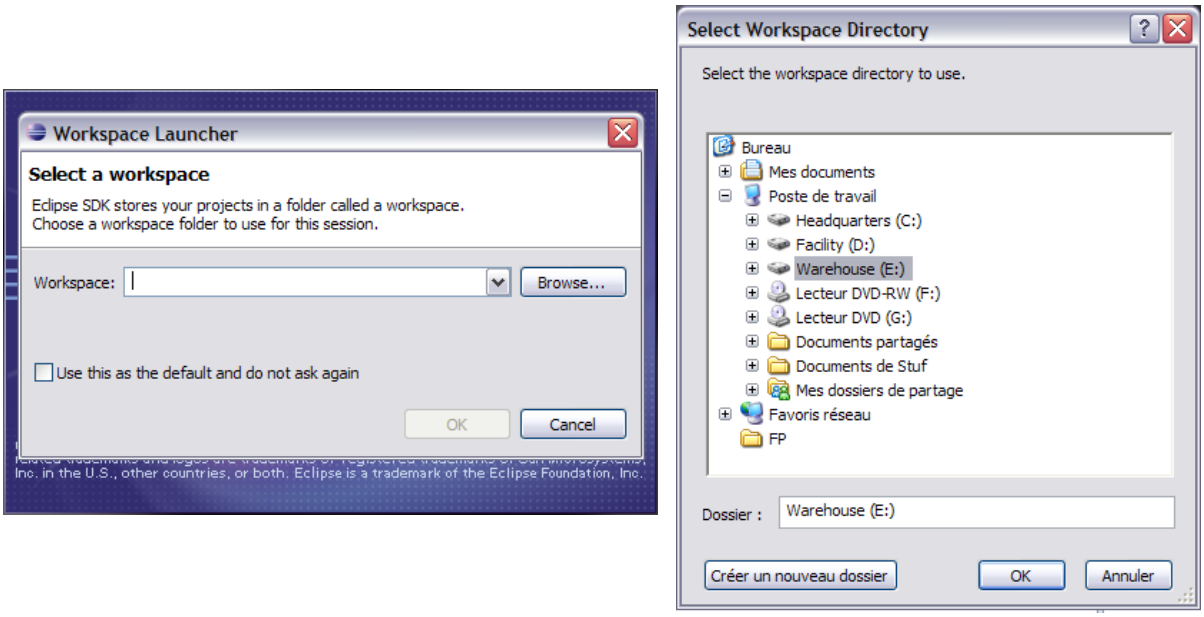

Figure 1 - Dialogue de sélection et de création d'un workspace.

Pour les enseignements de Java, nous allons créer un workspace spécifique lors du premier démarrage d'Eclipse. En effet, la liste de choix d'un workspace (image de gauche de la Figure 1) ne contient aucun item car c'est (normalement) la première fois que vous lancez Eclipse. Le bouton « Browse… » vous permet d'ouvrir un dialogue de parcours/sélection de fichier (image de droite) dans lequel vous allez pouvoir créer un répertoire.

A l'aide du dialogue de sélection de répertoires, créez le répertoire « MesProgrammesJavaS3 » (par exemple) sur votre lecteur personnel « H: ».

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 apparait 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, nous vous fournirons un projet que vous devrez compléter. C'est pourquoi nous allons voir la création, l'importation et l'exportation de projets.

Création d'un projet

On peut accéder au guide de création de projets par deux méthodes : par le menu « File/New/Project… » ou par le menu contextuel du le panneau « Package Explorer » (voir Figure 2).

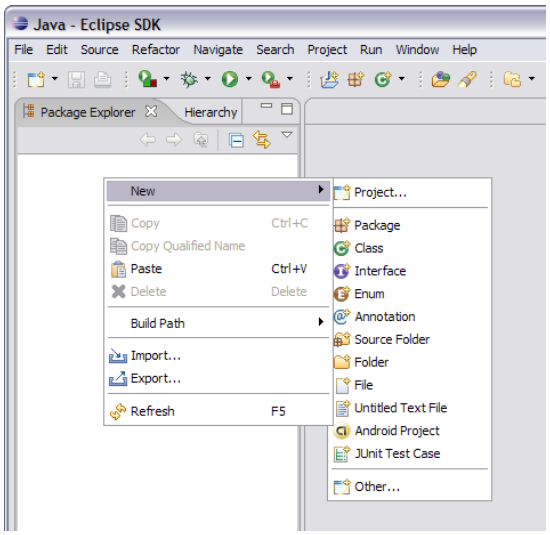

Figure 2 - Création d'un projet. Menu contextuel du panneau "Package Explorer".

Après avoir choisi « Project… », la fenêtre de création de projet (*wizard*) apparait (Figure 3a). Cette première étape permet de sélectionner quel type de projet l'on va créer.

Pour créer votre premier projet Java, choisissez « Java Project » et puis « Next ».

La seconde fenêtre (Figure 3b) permet de définir les propriétés essentielles du projet :

1. Son nom (ici ProjetTest). Ce nom sera aussi utilisé par Eclipse comme nom de

répertoire pour placer les fichiers du projet.

2. L'endroit où l'on veut créer le projet (« Contents »). Il est possible de créer un nouveau projet dans le workspace en cours (l'option choisie ici) ou de créer un projet Eclipse à partir de code source Java déjà existant.
3. 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. La structure du projet (« Project Layout »). Il y a deux manières de structurer les projets : placer tous les fichiers (sources et binaires) à la racine du projet ou séparer les sources et binaires dans des répertoires différents (« src » et « bin »). Ici, la deuxième option a été choisie car plus propre.

A partir de ces premiers paramétrages, il est possible de cliquer sur « Finish » et de finaliser la création du projet. Nous allons toutefois voir d'autres options qui peuvent s'avérer utiles pour la création et le paramétrage des projets.

Définissez les propriétés d'un projet « BonjourToutLeMonde » en utilisant les mêmes options puis « Next ».

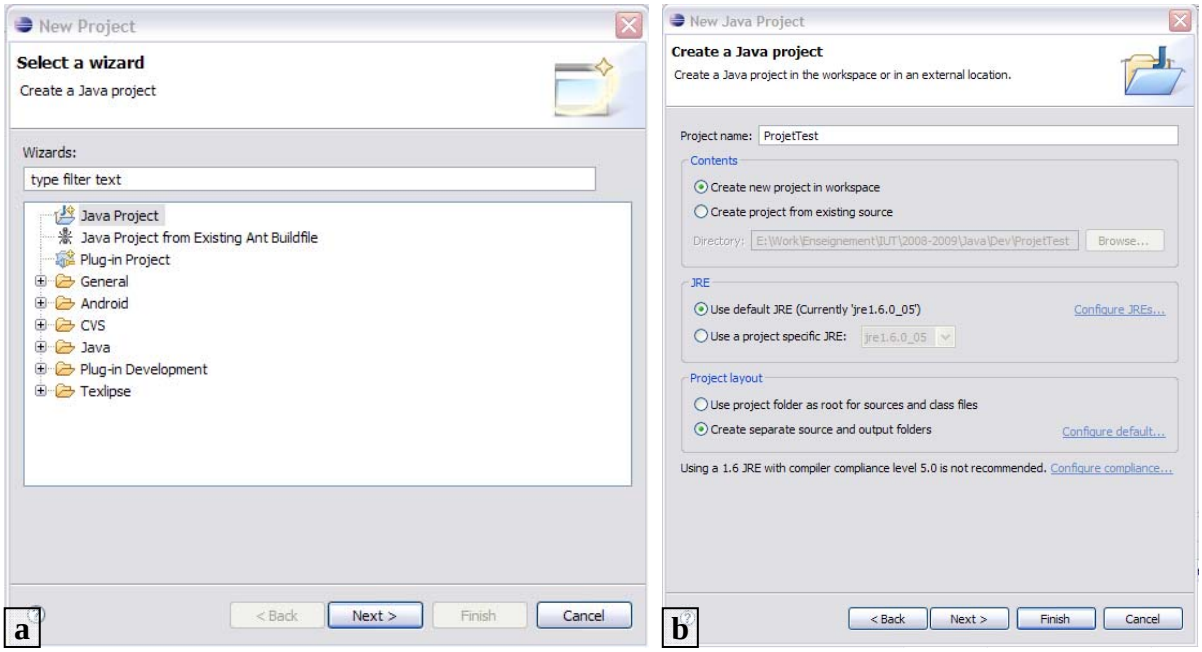

Figure 3 - Création d'un projet - Wizard (1).

La fenêtre suivante (Figure 4) permet de configurer des options plus avancées du projet. Le premier onglet (Figure 4a) permet de configurer les répertoires de sources du projet (fichiers .java). Le second onglet permet de spécifier des dépendances avec des projets (dans le cas où des classes d'un projet sont nécessaires pour en compiler et exécuter un autre). Nous ne nous attarderons pas sur ces options.

Le troisième onglet (« Libraries ») permet de définir des dépendances avec des librairies externes nécessaires pour compiler et exécuter le projet (« Java Build Path »). Nous allons voir comment utiliser ces options pour configurer notre projet afin qu'il utilise la librairie de tests unitaires JUnit.

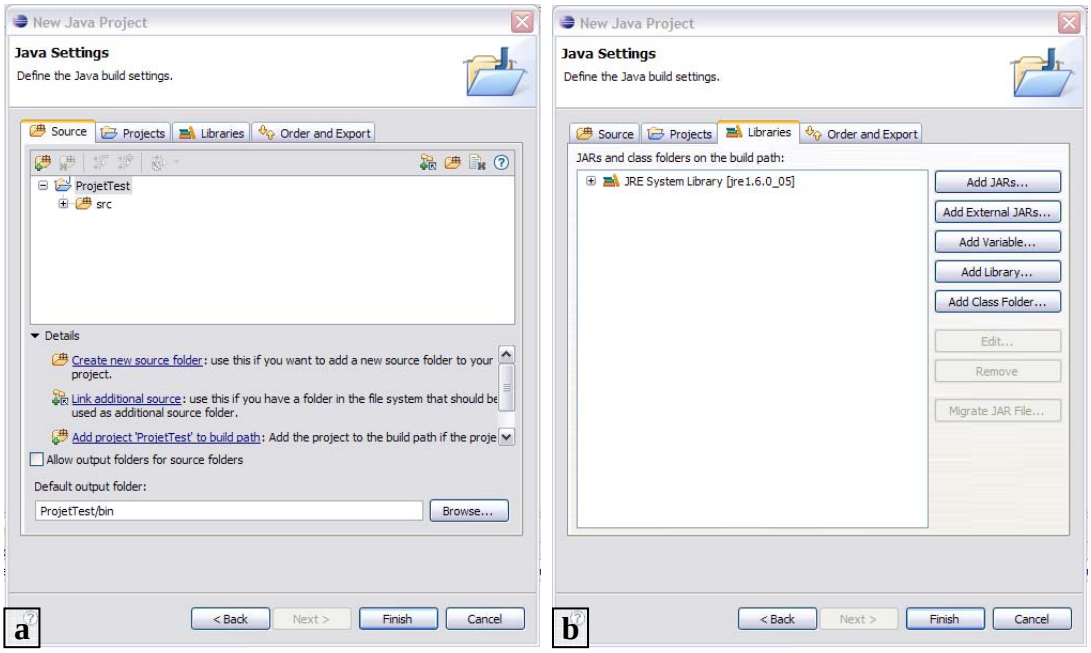

Figure 4 - Création d'un projet - Wizard (2).

Les boutons « Add Jars… » et « Add External Jars… » permettent de référencer des librairies qui n'ont pas encore été définies dans l'environnement d'Eclipse.
Le bouton « Add Library… » permet d'ajouter une librairie déjà référencée dans l'environnement Eclipse. Lorsque l'on clique sur ce bouton, la liste des librairies disponibles apparait (Figure 5a). Il suffit de choisir dans la liste et de cliquer sur « Next » (ici, l'on choisit JUnit). Dans le cas ou plusieurs versions ou options de la librairie sont disponibles, on peut choisir laquelle utiliser (Figure 5b). Enfin, lorsque l'on clique sur « Finish », la librairie est ajoutée aux dépendances et l'on revient aux options du projet (Figure 5c).

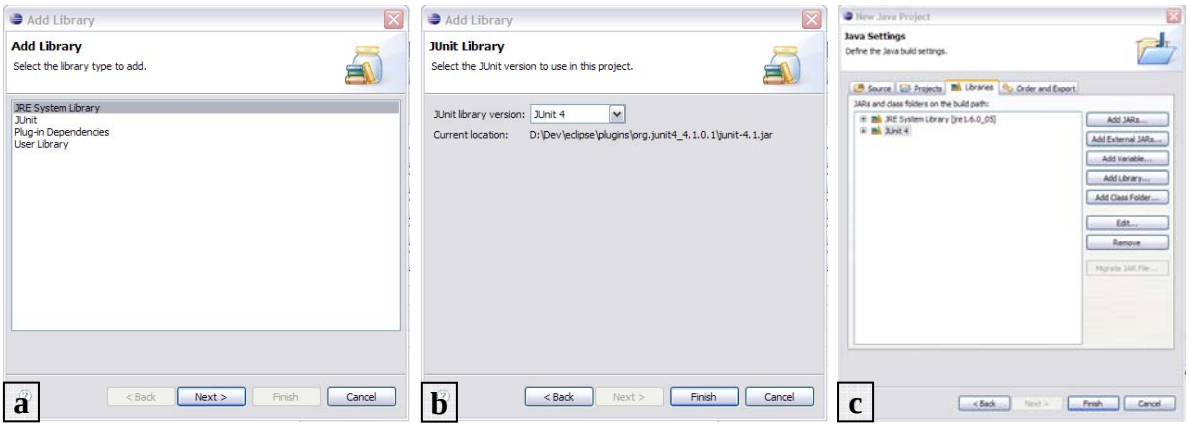

Figure 5 - Ajout d'une librairie aux dépendances du projet.

Enfin, lorsque l'on clique sur « Finish », le projet est effectivement créé dans le workspace et l'on revient à la vue globale d'Eclipse (Figure 6).

Ajoutez la librairie JUnit (4) à votre projet puis terminez sa création.

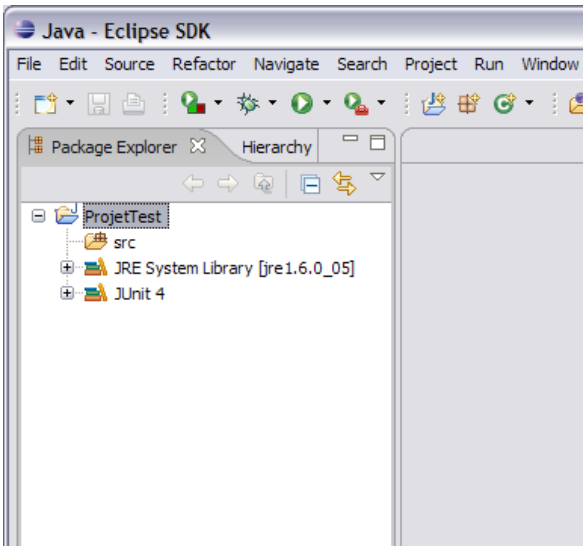

Figure 6 - Le projet "ProjetTest" est créé.

Configurer

Vous pouvez accéder aux options de configuration de votre projet par le menu « Project/Properties » ou par un clic droit sur votre projet dans l'onglet « Package Explorer » de la Figure 6. Cette boîte de configuration vous permet de paramétrer de nombreuses options (comme la version de Java à utiliser, les librairies à inclure, etc.) que nous ne détaillerons pas ici. A vous de les découvrir au fur et à mesure de votre utilisation d'Eclipse.

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 apparait (Figure 7). 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 : « BonjourToutLeMonde ». Une fois le nom saisi, le bouton « Finish » est activé et lorsque l'on clique dessus, le fichier « BonjourToutLeMonde.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 partie de code sont déjà écrites (Figure 7).

Créez la classe « BonjourToutLeMonde » dans le projet créé précédemment et recopiez-y le code donné lors du TP1a.

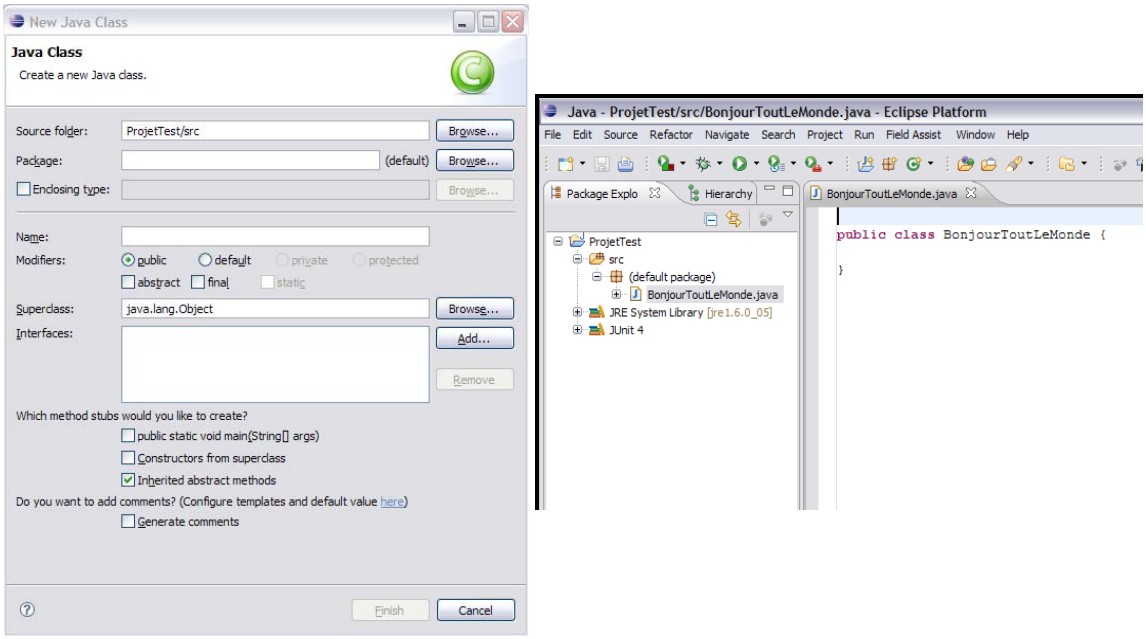

Figure 7 - Création d'une nouvelle classe.

Exporter/Importer un projet

Eclipse permet aussi d'exporter et d'importer des projets complets sous la forme de fichiers d'archives, de manière à les transférer dans d'autres workspaces ou sur d'autres machine. Nous utiliserons souvent cette possibilité au cours des TP de java pour vous fournir des projets à compléter. Vous pourrez aussi l'utiliser pour vous passer simplement vos projets entre binômes.

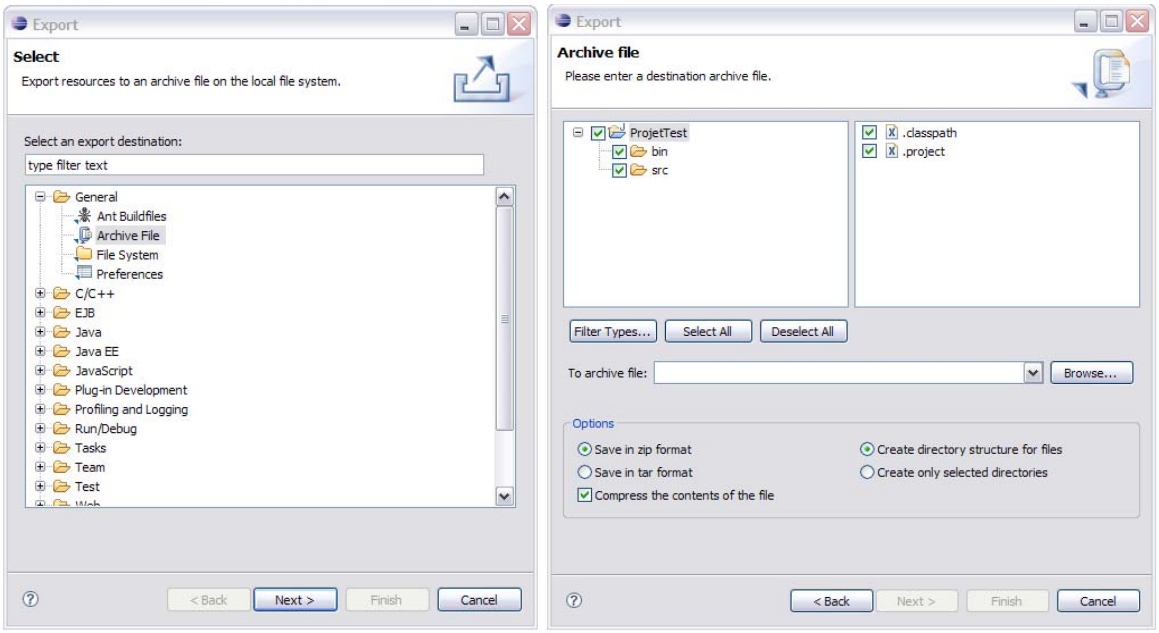

Figure 8 - Exporter un projet.

Pour exporter un projet, il suffit de sélectionner le projet et de choisir le menu « File/Export… » ou de faire un clic droit sur le projet et de choisir « Export… ». La fenêtre

de gauche de la Figure 8 apparait alors et permet de choisir quel type d'export l'on veut faire. Ici, nous choisissons « General/Archive File ». Après avoir cliqué sur « Next », la fenêtre de droite de la Figure 8 permet de choisir les options d'exportation (fichiers à exporter, nom du fichier d'export, format de l'archive). Par défaut, tout est sélectionné et c'est ce qu'il faut faire pour exporter un projet complet de manière à le re-importer dans un autre workspace. Lorsque tout est renseigné, il suffit de cliquer sur « Finish » pour créer le fichier archive.

Exportez la totalité du projet créé précédemment dans un fichier « zip » nommé « ProjetTest.zip ». Ouvrez cette archive avec le navigateur de fichiers de windows pour examiner son contenu.

L'importation d'un projet s'effectue à partir du menu « File/Import… » ou du menu contextuel de l'onglet « Package Explorer ». Une fenêtre similaire à la fenêtre de gauche de la Figure 8 apparait et permet de choisir le type d'import (Eclipse permet d'importer des projets entier, mais aussi des fichiers individuels, compressés ou non, etc.). Ici, nous choisirons par contre « Existing Projects into Workspace ». La fenêtre de gauche de la Figure 9 apparait alors. Il suffit de spécifier le fichier « zip » contenant le projet à importer dans le champ « Select archive file » et de cocher les projets à importer une fois qu'Eclipse a analysé l'archive. Si le fichier spécifié ne contient pas de projet, il n'est évidemment pas possible d'importer de projet…
Ici, nous voyons dans la Figure 9 l'importation du projet que l'on vous fournira comme base de travail pour le TP1c. Vous devrez réaliser ces opérations pour importer le projet dans votre workspace personnel. La fenêtre de droite de cette figure montre que le projet a bien été importé.

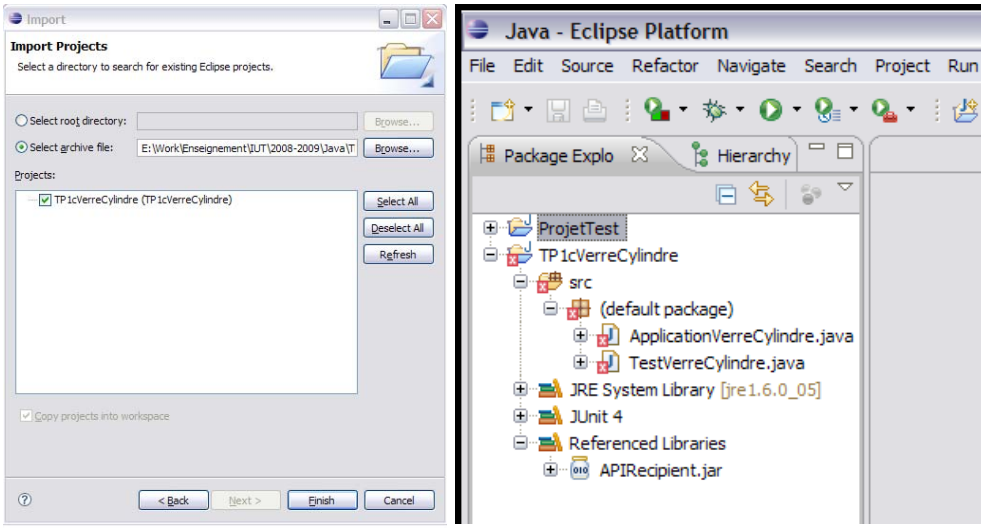

Figure 9 - Importer un projet.

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 (cas complexes qui nécessite des lignes de commandes longues et complexes ou des fichiers « batchs »).

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. Par exemple nous voyons dans la fenêtre de droite de la Figure 9 des croix rouges sur les fichiers que nous avons importés, signifiant que ces derniers contiennent des erreurs de compilation. Comme vous le découvrirez en utilisant Eclipse, ces erreurs sont mises en valeur et même expliquées dans l'éditeur de code.

Tests

Eclipse permet de lancer interactivement des tests JUnit que vous aurez écrits afin d'effectuer les tests unitaires des classes de vos projets. Cela vous sera expliqué en cours et en TD.

Exécution

Enfin, pour lancer l'exécution d'un programme que vous avez écrit, il suffit de faire un clic droit dans l'onglet « Package Explorer » sur la classe contenant la méthode « Main » de votre programme et de choisir « Run as/Java Application » ou de cliquer sur le bouton  dans la barre d'outils, après avoir sélectionné la classe contenant la méthode « Main ».

Lancez l'exécution du programme « BonjourToutLeMonde » que vous avez créé dans le projet « ProjetTest » de la première partie de ce tutoriel.

Suite…

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/

- Le tutoriel très complet de J-M DOUDOUX :
http://www.jmdoudoux.fr/java/dejae/indexavecframes.htm