

TP0 : Prise en main

15/09/21

Le but de cet exercice est de configurer (sous linux) l'environnement de programmation (Eclipse) qui sera utilisé en TP de Programmation.

Eclipse est un outil qui offre de nombreuses fonctionnalités utiles dans le cycle de développement d'un logiciel ; dans le cadre de nos TP, il nous servira principalement à exécuter la saisie de programmes en java et c++, à les compiler, les mettre au point et les exécuter.

Espace de travail : Avant de débiter, vous devrez créer (avec la commande *mkdir*) un répertoire de nom *prog* dans votre répertoire personnel ; il servira d' espace de travail (workspace) pour Eclipse : c'est là que vous placerez les différents TP dans des projets gérés par Eclipse.

1 Configuration d'Eclipse

Cette configuration est à faire une fois pour toutes ; lorsqu'il faudra saisir des informations au clavier, pensez à respecter la casse (majuscule/minuscule). Démarrez l'environnement de programmation Eclipse.

1.1 Espace de travail par défaut

Vous allez modifier l'espace de travail par défaut utilisé par Eclipse :

- Au démarrage, Eclipse affiche une boîte de dialogue de titre "Workspace Launcher" ;
- cliquez "Browse" pour naviguer dans les répertoires ; ceci ouvre une fenêtre de titre "Select Workspace Directory" ; sélectionnez le répertoire *prog* créé au début ;
- cochez la case "Use this as the default..." ;
- validez ("OK") ; Eclipse finit de démarrer.

1.2 Préférences générales

Ouvrez le panneau des préférences (menu "Window/Preferences") ; dans la partie gauche de ce panneau se trouvent des rubriques ; celles qui ont une flèche ou un + à leur gauche comportent des sous-rubriques, qu'on peut afficher en cliquant sur la flèche ou le + ; dans la partie droite sont affichées des options propres à la rubrique sélectionnée.

1.2.1 Enregistrement automatique

Sélectionnez "General/Workspace/Build" :

- 1) cochez la case "Save automatically before build" (oblige Eclipse à enregistrer vos fichiers avant toute compilation) ;
- 2) dé-cochez la case "Build automatically" (empêche Eclipse de recompiler votre programme lors de chaque sauvegarde) ;

1.2.2 Compilateur

- 1) sélectionnez "Java/Compiler" ; fixez le niveau de compatibilité du compilateur à 1.8 ("Compiler compliance level") ;
- 2) sélectionnez "Java/Compiler/Errors" puis ouvrez la rubrique "Deprecated and Restricted API" puis, en face de la sous-rubrique "Forbidden reference", sélectionnez "Ignore" au lieu de "Error" ;
- 3) cliquez "Apply".

1.2.3 Perspective de travail

Une perspective de travail est composée d'un certain nombre de fenêtres et d'onglets qui servent à présenter de façon structurée les informations d'un projet.

Ouvrez la rubrique "General/Perspectives" : dans la sous-rubrique "Open the associated perspective when creating a new project", cochez la case "Always open"

1.2.4 Divers

- Ouvrez la rubrique “General/Editors/Text Editors” et cochez la case “Show line numbers” ;
- Ouvrez “General/Editors/Text Editors /Spelling” et dé-cochez la case “Enable spell checking”.
- Validez autant de fois que nécessaire.

Remarque : inutile de sauvegarder les préférences sur le serveur eclipse.org...

La configuration est terminée ; vous pouvez fermer le panneau “Welcome” en cliquant sur la croix à droite de l’onglet : l’interface de travail d’Eclipse s’affiche.

2 Création d’un projet

Vous allez maintenant créer un projet java, y intégrer un fichier source java, le compiler et l’exécuter ; ceci vous permettra de valider votre configuration ; à chaque étape, si vous n’obtenez pas les informations correctes, vérifiez la configuration de l’élément erroné et refaites les opérations nécessaires.

Créez un nouveau projet java (File/New/Project... Java Project) ; donnez-lui un nom (par exemple : *tp0*) ; vérifiez, dans le champ “Location”, que le projet que vous créez est bien situé dans le répertoire créé tout au début de la configuration (section 1.1) ; si tout va bien, cliquez “Finish”.

L’explorateur de paquetages (onglet “Package Explorer”) s’affiche sur la gauche : il permet de naviguer dans les différents fichiers d’un projet java.

2.1 Incorporation d’un fichier source dans le projet

Pour incorporer un fichier source dans votre projet, il suffit de copier coller le fichier source (**.java**) dans le bon repertoire. Par exemple, télécharger le fichier **Exemple.java** depuis moodle et copier-le dans le sous-répertoire *src* de votre projet.

2.2 Explorer le fichier


À l’aide de l’explorateur de paquetages, naviguez à l’intérieur du répertoire *src* , puis dans le paquetage par défaut (“default package”) : vous devriez voir le fichier *Exemple.java* ; cliquez sur la flèche à sa gauche : la liste qui s’affiche vous montre la structure interne du fichier *Exemple.java*, constitué d’une classe (**Exemple**), elle-même composée de plusieurs fonctions : *afficherTableau*, *indiceMax*, *initialiserTableau*, *main*, *trierNombres* ;

Faites un double-clic sur la fonction *trierNombres* : ceci ouvre le fichier *Exemple.java* dans un éditeur qui occupe la fenêtre centrale de votre environnement de développement, et sélectionne la fonction *trierNombres* ; faites un double-clic sur un autre nom de fonction (*main* , par exemple) : l’éditeur vous affiche la fonction sélectionnée : c’est un moyen rapide de naviguer directement d’une fonction à une autre au sein de votre fichier.

À tout instant, vous pouvez agrandir un onglet en double-cliquant sur son titre et le ramener à sa taille initiale en faisant de même.

2.3 Compiler et exécuter

- Assurez-vous de sélectionner le fichier (*Exemple.java*) dans l’explorateur puis compilez-le (menu “Project/Build Project”) ; corrigez les erreurs signalées par eclipse puis recompilez.
- Pour exécuter le programme depuis Eclipse , menu “Run/Run As/Java Application ” : ceci va démarrer le programme et ouvrir l’onglet “Console” ; c’est dans cet onglet que se fera l’interaction avec le programme : saisissez-y les données demandées par le programme (validez chaque nombre par l’appui de la touche “Entrée”) ; n’hésitez pas à agrandir la console (double-clic sur l’onglet).
- Le programme devrait afficher la suite de nombres telle que vous l’avez saisie, puis la même suite triée par ordre croissant.
- Fermez Eclipse (menu “File/Exit”).
- Redémarrez Eclipse : vous devriez retrouver le projet dans l’état précédent ; si ce n’est pas le cas, c’est probablement que vous n’utilisez pas le bon espace de travail ; changez-en ainsi :

- 1) menu “File/Switch Workspace” ; choisissez le répertoire créé en section 1.1.
 - 2) après le redémarrage d’Eclipse, faites ce qui est décrit en section 1.1.
- Vous pouvez à nouveau exécuter le programme ; cette fois-ci, il suffit de cliquer l’icône  ceci va démarrer le dernier programme exécuté.

3 Implémentation Racine

Dans cette section, on vous propose d’implémenter une fonction **racineCarree** selon la spécification suivante :

```
1  /**
2   * Calculer une valeur approchée de la racine carrée d’un nombre
3   * @param r : réel dont on veut calculer la carrée r >= 0
4   * @param epsilon : précision du calcul epsilon > 0
5   * @return un réel a, valeur approchée à epsilon près de racine_carrée de r
6   */
7  public static double racineCarree(double r, double epsilon) {
8      // vous devez implémenter cette fonction
9      // Implémenter des conditions sur la validité des paramètres : r >= 0 et epsilon
10     // si les conditions initiales ne sont pas respectées, votre programme devra
11     // retourner une erreur
12 }
```

Un algorithme très simple est décrit sur la page https://fr.wikipedia.org/wiki/Méthode_de_Héron : c’est un cas particulier de la “Méthode de Newton” appelée “algorithme de Babylone” ou “méthode de Heron”. Programmez ensuite la fonction **main** qui fait appel à la fonction **racineCarree** ; faites un test en fournissant dans l’instruction d’appel des valeurs pour r et epsilon qui respectent la spécification de la fonction.

4 Test Unitaire

Consultez le document “Test unitaire avec JUnit”.

Copiez le fichier *TestUnitaireRacine.java* dans le répertoire de votre projet. C’est un programme de test unitaire que vous devez utiliser pour tester la fonction *racineCarree*.

Si vous rencontrez des erreurs lorsque vous compilez le programme de test, vérifiez d’abord si vous avez fait ce qui est indiqué dans le document ci-dessus ; vérifiez ensuite que votre fonction respecte bien la spécification imposée.

Votre fonction est considérée comme testée (mais pas obligatoirement correcte), si elle passe avec succès tous les tests.

5 Création d’un projet “from scratch”

Pour créer un projet “à partir de rien”, il faut :

- 1) créer un nouveau projet (voir section 2)
bibliothèque : si besoin, associer une bibliothèque à votre projet :
 - soit lors de la création du projet ;
 - soit a posteriori : menu “Project/Properties”, “Java Build Path”, onglet “Libraries”, rubrique “Class-Path” ;
 - dans les deux cas :
 - “Add External Jars” : naviguez dans le système de fichiers jusqu’à l’emplacement du fichier jar qui contient la bibliothèque (par exemple : */share/esir1/prog/bibliotheques/types.jar*)
 - faites de même pour chaque bibliothèque requise

test unitaire : si besoin, ajouter la bibliothèque Junit4 au “classpath” de votre projet ; voir plus haut.

- éventuellement créer un ou plusieurs paquetages (dépendra du problème) (“File/New/Package”)
- créer une ou plusieurs classes (“File/New/Class”)
- programmer puis compiler chaque classe comme dans la section 2.

6 Conclusion

Vous venez de configurer l’environnement Eclipse : si vous avez effectué cette configuration avec soin, vous pourrez travailler dans de bonnes conditions lors des TP.

Vous avez aussi effectué les opérations élémentaires du cycle de vie d’un programme ; conservez ce document avec vous lors des TP et consultez-le quand ce sera nécessaire.