NFA031 (en journée) : TP 1 : prise en main de l'environement de travail JDK

V. Aponte

8 octobre 2018

Dans ce Tp nous abordrons les outils du JDK bruts et nous vous conseillons de procéder ainsi pendant 4 séances **avant ou en parallèle** à l'utilisation d'un *environnement de développement intégré* (IDE) plus riche et plus agréable (p.e. Eclipse).

Ce que vous devez faire la 1ère fois

1. Connectez vous sous Linux. Pour cela il peut être necessaire de demarrer ou redemarrer votre poste de travail. Une fois un système Linux en place :
   (a) à l'invitation `login` : tapez soit votre nom de login si vous avez une carte d'élève du Cnam, soit le login générique : `licencep`
   (b) à l'invitation `passwd` : tapez votre mot de passe (connectez vous sur le site du CNAM-DSI pour l'avoir), ou le mot de passe des login génériques (demandez à votre enseignant).
   Attention : les lettres doivent être tapées en mode majuscules. Lorsque vous tapez votre mot de passe, il n'y a pas d'écho de votre frappe à l'écran, cela protège votre mot de passe cotre les regards indiscrets.
2. Mise en place de l'environnement de travail :
   (a) récupérez dans le site du cours (onglet « Outils » le fichier `Terminal.java` `http://deptinfo.cnam.fr/Enseignement/CycleA/APA`. Utilisez pour cela un navigateur. Si vous avez des difficultés pour récupérer ce fichier, demandez l'aide de l'enseignant. Cet opération ne doit se faire qu'une seule fois, lors de la première séance de TP et non pas à chaque séance.

Utilisation d'un éditeur de texte + compilation + exécution

Exercice 1 : utiliser JDK depuis la console (commandes `javac` et `java`)

Il faut commncer par taper le programme dans un éditeur de texte. Un éditeur de texte est un logiciel qui permet de créer des fichiers textes. Ils ont des fonctions communes avec des traitements de texte, par exemple la recherche/remplacement, le copier/coller. Mais ils ont deux grosses différences : ils ne permettent pas de spécifier une mise en page avec par exemple des changements de police, du gras, du souligner ; ils sauvent les fichiers sous forme de texte seul, alors que les traitements de texte ont leur propre format (par exemple de .doc de word, qui est illisible pour un éditeur de texte).

Créer son programme source avec un éditeur de texte

Sous Windows, il faut utiliser un éditeur de texte qui ne met pas d'office l'extension .txt aux fichiers textes, puisque nos programmes Java doivent avoir l'extension .java. Parmis les possibilités en standard : le bloc-note qui est une application Windows et `Edit` qui est une commande qu'on utilise dans une fenêtre `Dos-invite de commande`. D'autres editeurs sont bien préférables, mais ils ne sont pas disponibles dès l'installation de Windows. Dans la plupart des cas, il faut les installer.

Sous Linux en salle de Tp : vous utiliserez `jedit`, un éditeur gratuit déjà intallé sur les machines de salles Tp (vous pouvez l'installer chez vous sous Linux ou MacOS :
`http://www.jedit.org/`http ://www.jedit.org/).

Quelque soit votre éditeur de texte, il faut y taper son programme. Pour les programmes des notes de cours, plutôt que de les retaper, il faut les copier/coller. Cela permet d'éviter les fautes de frappe et de gagner du temps. **Votre fichier doit avoir le même nom que la classe java qu'il contient** et l'extension `.java`. Attention : les majuscules et les minuscules doivent être distinguées. C'est à dire que si la classe s'appelle `Premier` le fichier doit s'appeler `Premier.java` et non `conversion.java` ou `PREMIER.java`.

Utiliser l'éditeur Jedit

Vous allez maintenant réaliser cette exercice avec un éditeur adapté à Java : JEdit. Pour cela, vous chercherez cette application dans le menu des programmes de votre poste et vous le lancerez. Vous pouvez regarder la vidéo qui se trouve sur le site du cours où vous trouverez également des instructions d'installation. Vous pourrez ensuite procéder à l'édition du programme correspondant au texte suivant :

```
/**
 * Un premier programme Java.
 * @author METTEZ ICI VOTRE NOM
 * Ce programme affiche "Mon programme no. 1"
 */

public class Premier {
  public static void main (String [] args){
     int n=1;
     System.out.println("Mon programme no. "+n);
  }
}
```

Compiler et exécuter

Ensuite le fichier que vous avec créer doit être compilé en utilisant la commande `javac`. Une fois ce programme compilé (s'il n'y a pas d'erreur signalés), on pourra l'exécuter en utilisant la commande `java`. On compile le programme une fois et on l'exécute autant de fois qu'on veut sans avoir à recompiler à chaque fois. En revanche, il faut recompiler après chaque modification du programme.

1. Compilation : dans la fenêtre Unix, tapez la commande `javac Premier.java`
   S'il n'y a pas de messages d'erreur, la compilation a reussi. Le fichier `Premier.class` est créé suite à la compilation. Vous pouvez vérifier cela en tapant la commande `ls Premier*`

2. Exécution : tapez dans la fenêtre Unix la commande `java Premier`
   Vous devez voir s'afficher à l'écran `Voici mon programme no. 1`

Ce que vous devez faire les autres fois

1. Se connecter sous Linux, suivant la procédure décrite plus haut
2. Lancer un éditeur de texte (ou Eclipse)
3. Ecrire les programmes correspondant aux exercices de la séance, et les sauvegarder dans des fichiers de même nom que la classe principale, suivi de l'extension `.java`
4. Compiler ces programmes avec la commande `javac` suivie du nom de votre fichier (Compilation). Le cas écheant, corriger les erreurs signalés par le compilateur (Mise au point) ;
5. Une fois la compilation réussie, exécutez ces programmes en tapant (dans la fenêtre Unix) la commande `java` suivie du nom de la classe principale (Exécution)
6. N'oubliez pas de sortir de toutes les applications et de vous déconnecter (Déconnexion).

Exercice 1 : comprendre les erreurs de compilation

Revenez sur le code source de votre programme `Premier.java`. Vous y allez introduire une erreur de syntaxe : à la ligne 4, remplacez le premier mot réservé `public` par `publique`.

— Après remplacement, le nouveau mot n'est plus coloré en bleu, comme tous les autres mots clés.
— Recompilez votre programme (commance javac) ; une erreur s'affiche avec (à peu près) ce texte :

```
1 error found:
4: Premier.java
Error:  class, interface, or enum expected
```

Cherchez à comprendre ce que ce message d'erreur vous indique :
— le nom et chemin complet du fichier,
— le numéro de la ligne,
— à cette ligne, le détails sur l'erreur trouvée.

Exercice 2 : lire avec `Terminal`

Dans cet exercice vous aurez besoin de la classe `Terminal.java` que vous trouverez dans la dans la rubrique *Ressources* sur le site du cours. Vous devez recupérer ce fichier et l'enregistrer dans le répertoire où se trouvent tous vos programmes. Tapez dans un nouveau fichier le code source suivant :

```
/**
 * Lit deux entiers au clavier et calcule puis
 * affiche leur addition et multimplication.
 * @author V. Aponte
 */
public class DeuxInt{
  public static void main(String[] args){
    System.out.println("Entrez un entier: ");
    int x = Terminal.lireInt();
    System.out.println("Entrez un entier: ");
    int y = Terminal.lireInt();
```

```
    int somme = x+y;
    int mult= x*y;
    System.out.println("La somme des deux nombres "+x+" et "+y+" est "+somme);
    System.out.println("La multiplication des deux nombres "+x+" et "+y+" est "+mult);
  }
}
```

Ce programme lit au clavier deux nombres entiers, calcule leur somme, leur multiplication et affiche ces résultats.

Question 1

1. Sur une feuille de papier, indiquez le type de construction java se trouvant dans chaque ligne : commentaire, affichage, déclaration de variable, lecture avec `Terminal`, opération arithmétique, affectation, autre. Dans une même ligne il peut y en avoir plusieurs. Par exemple en ligne : `int somme = x+y;` il y a une déclaration de variable, une opération arithmétique, et une affectation.
2. Essayez de prévoir ce que va exécuter chaque ligne de ce programme, et sur une feuille de papier retracez ce qu'il devrait afficher, pas à pas.
3. Compilez ce ficher.
4. Sur une feuille de papier, composez une table avec divers cas de tests pour ce programme. Pour mémoire, vous devez prévoir des valeurs différentes pour les entrées du programme et les résultats attendus pour ces entrées.
5. Exécutez ce programme en lui « donnant » les entrées de votre jeu de tests, et vérifiez si les réponses obtenues sont celles que vous attendiez.