



## ALG2ir-TD01 : Environnement de développement et rappels Java

### 1 Présentation du TD

En première année, vous avez étudié Java comme premier langage de programmation. Depuis votre arrivée en deuxième année, le principal cours de programmation qui vous est délivré couvre les langages C et C++, tandis que vos connaissances en Java ne sont plus développées ni entretenues. Un des but de ce travail dirigé est de vous rafraîchir la mémoire pour ce qui concerne le langage Java<sup>2</sup>.

D'autre part, en 1<sup>re</sup> année, votre apprentissage de Java s'est accompli dans un environnement de programmation rudimentaire, dans le but pédagogique, entre autre, de ne rien vous masquer des étapes entre l'édition d'un source et l'exécution du programme résultant. Il existe cependant des environnement de développement intégrés (EDI, IDE en anglais) facilitant la vie du développeur. Un second objectif de ce TD est de vous guider lors de vos premiers pas dans l'EDI choisi, à savoir NetBeans<sup>3</sup>.

### 2 NetBeans 5.0

NetBeans est un EDI adapté à Java, gratuit et *open source* (initialement développé par SUN lui-même). Nous allons l'utiliser dans le cadre des Ateliers Logiciels (ALG2ir). Notez cependant tout de suite que nous ne l'emploierons pas dans toute sa puissance. Ainsi, par exemple, lorsque nous développerons des applications à interface graphique, nous ne recourrons pas à l'outil graphique de développement offert par NetBeans ! Nous y reviendrons par la suite. . .

#### 2.1 Apprentissage

On trouve en ligne, à l'adresse <http://www.netbeans.org/kb/55/quickstart.html>, un guide de prise en main de NetBeans 5.5, parfaitement adapté à la version 5.0. Nous vous encourageons vivement à suivre ce tutoriel.

---

<sup>1</sup> Le texte de ce TD s'inspire très largement de ceux des Ateliers Logiciels Java (2<sup>e</sup> Gestion et 3<sup>e</sup> Industrielle et Réseaux) créés principalement par MCD et VAK, mais auxquels ont également collaboré RFS et SMB (et peut-être d'autres que j'ignore, qu'ils me pardonnent).

<sup>2</sup> Remarquez que la version de Java disponible actuellement à l'école sur les machines MS-WINDOWS 2000 n'est pas la version 1.6. Nous utiliserons la version 1.5 dans le cadre du cours de ALG2ir.

<sup>3</sup> Remarquez que la version de NetBeans disponible actuellement à l'école est la version 5.0. Cette version est téléchargeable à l'adresse <http://www.netbeans.info/downloads/index.php?rs=2>.

NetBeans 5.0 lui-même est pourvu d'un tutoriel de démarrage fort bien fait, disponible depuis l'application NetBeans elle-même<sup>4</sup> ou sur le site du projet<sup>5</sup>.

Suivez le tutoriel afin d'être en confiance avec cet environnement. Soyez particulièrement attentifs aux points :

- 1 : « Setting Up a Project » ;
- 2 : « Creating and Editing Java Source Code » ;
- 3 : « Refactoring » ;
- 4 : « Compiling and Running a Project » ;
- 5 : « Testing and Debugging a Project », ignorez les sous sections relatives aux tests unitaires mais allez directement à « Debugging a project ».

## 2.2 Avantages et inconvénients de l'utilisation d'un EDI

Comme vous venez de le constater, NetBeans offre de multiples services très intéressants tels que la coloration contextuelle, l'auto-complétion, la recherche dans la documentation Javadoc, le débogage, etc.

Dans le cadre d'un usage pédagogique, certaines des aides proposées par NetBeans vont un peu *trop loin*. Par exemple, la génération de code par NetBeans (hormis les modèles, *templates*) ne sera pas exploitée dans le cadre des ALG2ir. Ainsi, pour mettre en page une application à interface graphique ou pour associer l'exécution d'une méthode à l'occurrence d'un événement, nous ne recourons pas aux interfaces graphiques et *wizards* de NetBeans, mais coderons tout à la main...

Un autre point faible, pédagogiquement parlant, lors du développement à l'aide de NetBeans, est le masquage du `PATH` et du `CLASSPATH` ainsi que la gestion automatique de la correspondance entre le paquetage (package) d'une classe et l'emplacement dans l'arborescence des dossiers du fichier où cette classe est définie. Gardez à l'esprit que vous devez toujours être capable de :

- définir correctement les variables d'environnement `PATH` et `CLASSPATH`<sup>6</sup> ;
- sauver vos sources au bon endroit (*cf.* `CLASSPATH` et paquetage) ;
- compiler vos sources ;
- exécuter vos byte codes ;

*hors* NetBeans, via la ligne de commande, comme on vous l'a appris en 1<sup>re</sup> année !

## 3 Java

Voyons Java à l'aulne de quelques-unes de ses différences avec le langage C++ que vous découvrirez cette année :

- il n'y a pas de précompilateur en Java ;

---

<sup>4</sup> Menu `Help` >> `Option Tutorials` >> Sous-option `Quick Start Guide`, puis, sur la page html ouverte, cliquer sur le lien nommé *General Java applications or libraries*, tout se passe localement, pas de connexion internet nécessaire.

<sup>5</sup> À l'adresse <http://www.netbeans.org/kb/50/quickstart.html>.

<sup>6</sup> Pas de « " » dans la définition de la valeur de `CLASSPATH` !

- en C++, il y a trois manières d'« accéder par le biais d'une variable » à un type primitif ou à un objet : directement, par le biais d'un pointeur ou d'une référence (qu'en est-il en Java ?);
- en Java, (presque) tout est OO (qu'est-ce qui ne l'est pas ?);
- un ramasse-miettes (*garbage collector*) existe en Java, des destructeurs apparaissent en C++, comment la mémoire est-elle gérée dans les deux langages ?;
- pas de pointeur en Java et pourtant tous les objets sont dynamiques;
- les tableaux aussi sont des objets et sont dynamiques en Java (qu'est-ce que ça veut dire exactement ? leur taille peut-elle varier ?);
- un tableau à deux dimensions est vraiment un tableau de tableaux en Java (en C++, ça marche comment ?);
- les arguments de fonction (ou méthode) peuvent être transmis par valeur ou par référence en C++ (et en Java ?);
- C++ est polymorphe ou non, selon les désirs du programmeur (et Java ?);
- une classe Java peut être abstraite (y-a-t-il un équivalent en C++ ?);
- une classe *complètement* abstraite, en Java, est une « interface » (*quid* en C++ ?);
- il n'y a pas d'héritage multiple en Java, mais une classe peut « implémenter » plusieurs interfaces;
- toute classe Java hérite (explicitement ou implicitement) de la classe `Object` (directement ou indirectement);
- à partir de sa version 1.5, Java introduit la notion de généricité, cette notion existe également en C++ par le biais des *templates*, les techniques mises en œuvre en Java et C++ sont-elles semblables ?;
- le « *Collection Framework* » offre ce qu'il faut pour manipuler des collections, notamment la classe `List` (y-a-t-il un équivalent C++ ?);
- un système de documentation est prévu en Java (y-a-t-il un équivalent C++ ?);
- des bibliothèques (API) pour le développement d'*applications à interface graphique* font partie des bibliothèques standard de Java (y-a-t-il un équivalent C++ ?);
- des mots-clés Java et des bibliothèques (API) standard pour le développement d'*applications à plusieurs fils d'exécution (multi-thread)* existent (y-a-t-il un équivalent C++ ?);
- des bibliothèques (API) pour le développement d'*applications réseaux* font partie des bibliothèques standard de Java (y-a-t-il un équivalent C++ ?);
- etc.

Pour vous rafraîchir davantage la mémoire, n'hésitez pas à consulter le cours de 1<sup>re</sup> année, la Javadoc et des ouvrages de référence. Par ailleurs, les quelques exercices suivants vont vous aider dans cette tâche.

## 4 Exercices

Lors de la résolution de tous les exercices de ce TD et de ceux à venir, veillez à ne pas travailler hors paquetage, mais définissez proprement une arborescence de paquetages personnelle et propre au cours ALG2ir. Par exemple, mes classes pour le premier exercice de ce TD se trouvent dans le paquetage `nvs.alg2ir.td01rappel.trianglepascal`.

**Ex1.** Écrivez une fonction qui crée un triangle de Pascal de  $n$  lignes puis une fonction qui

l'affiche. Écrivez enfin une fonction principale pour tester le tout.

Le paramètre  $n$  est récupéré par la ligne de commande. Le programme affiche un petit message de type « `usage : ...` » si il est appelé avec un mauvais nombre d'arguments ou avec un argument non entier.

Pour rappel, le triangle de Pascal,  $P$ , de  $n$  lignes est défini par la récurrence :

$$\begin{aligned} P(i, 1) &= P(i, i) = 1 \quad \text{pour } i = 1, 2, \dots, n, \\ P(i, j) &= P(i - 1, j - 1) + P(i - 1, j) \quad \text{pour } 1 < j < i. \end{aligned}$$

**Ex2.** Récupérez dans mon eDistri (nvs, ou celui de votre professeur) le programme « Répertoire », c'est-à-dire les trois fichiers `Personne.java`, `Répertoire.java` et `TestRépertoire.java`, et servez-vous en pour créer un projet NetBeans.

Les sources fournis ont été développés *avant* le JDK 1.5. Avant toute autre chose, mettez-les à jour !

La documentation de la méthode `add` de la classe `Répertoire` indique :

*TODO : L'ajout ne se fait que si la personne n'y est pas encore.*

Réalisez ce désir.

Ce problème étant réglé, modifiez ce qui doit l'être de sorte à pouvoir associer plusieurs numéros de téléphone à une même personne.

Veillez à utiliser au mieux les possibilités du langage.