

Programmation en JAVA

Année Universitaire 2016-2017

Introduction

Origine de Java

- **1991** : Projet Green Project (Sun Microsystems) pour proposer un nouveau langage de programmation permettant le développement d'applications portables pour des appareils électroménagers, ainsi qu'un environnement d'exécution pour leur exécution.
- **Objectif** : rendre les appareils électroménagers mieux contrôlables, plus interactifs et capable de communiquer entre eux. Tous ces appareils devant être contrôlés par l'intermédiaire d'une télécommande universelle qui reposait sur l'environnement d'exécution (un mini système d'exploitation) permettant l'exécution des programmes et la communication avec l'ensemble des appareils.

Origine de Java

- Maître d'œuvre du projet, James Gosling voulait créer un langage orienté objet qui reprend les principales caractéristiques du langage C++, tout en modernisant son utilisation (suppression du pré-processeur, renforcement du contrôle de typage, ...).
- Le langage était initialement baptisé Oak (chêne)
- La principale force de ce langage est sa portabilité :
 - La taille imposée des types de données permet de rendre les programmes plus portables et simplifier leur intégration dans n'importe quel type de matériel (processeur 16 bits, 32 bits, ...).
 - La génération d'un code machine pouvant être exécuté sur n'importe quel type de plates-formes.

Qu'est ce que Java ?

Définition de SUN :

"Java : a simple, object-oriented, distributed, robust, secure, architecture neutral, portable, high-performance, multithreaded, and dynamic language"

"Java est un langage simple, orienté objet, distribué, robuste, sûr, indépendant des architectures matérielles, portable, de haute performance, multithread et dynamique"

Qu'est ce que Java ?

Pour **Hortsmann** et **Cornell**, java est plus qu'un langage, c'est une *"plate-forme complète, disposant d'une importante bibliothèque, d'une grande quantité de code réutilisable et d'un environnement d'exécution qui propose des services tels que la sécurité, la portabilité sur les systèmes d'exploitation, et le ramasse-miettes automatique ... C'est un langage à la syntaxe agréable et à la sémantique compréhensible ..."*

(Livre : Au cœur de Java – Pearson Education - 2008)

Qu'est ce que Java ?

- Langage incontournable actuellement, très utilisé, notamment par les programmeurs professionnels
- Langage de **programmation orienté objet**
- Emprunte sa syntaxe en grande partie aux langages C/C++. Il a repris les principales caractéristiques de C++, en éliminant ses points difficiles, tout en étant moins encombrant et plus portable (pour pouvoir être intégré dans n'importe quel appareil...).

Qu'est ce que Java ?

- Java est un langage compilé: avant d'être exécuté, il doit être traduit dans le langage de la machine sur laquelle il doit fonctionner
- Les programmes Java peuvent être exécutés sous forme d'applications indépendantes, ou distribuées à travers le réseau et exécutées par un navigateur Internet sous forme d'applets
- Langage approprié pour le Web : programmes de faibles taille, portables (capable de fonctionner sur toutes les machines différentes d'un même réseau) ...

Bref historique (voir Doc1)

- **1991** : Sun Microsystems lance le projet "Oak" (chêne) pour anticiper l'évolution de l'informatique
- **1995** : Lancement officiel de Java 1.0 à la conférence SunWorld
- **1996** : Lancement du JDK 1.0.1 (*Java Development Kit*)
- **1998** : lancement de J2SE 1.2 (Java 2 Standard Edition)
- **1999** : lancement de J2EE
- **2002** : lancement de J2SE 1.4
- **2004** : lancement de J2SE 5.0
- **2006** : Passage sous licence GPL: logiciel libre
- **2009** : Rachat de Sun par la société Oracle.
- **2011** : Java SE 7
- **2014** : Java SE 8

Les différentes éditions de Java

- Java présente 3 éditions différentes qui regroupent des APIs par domaine d'application :
 - **JSE (Java Standard Edition)** : C'est l'édition standard objet de ce cours. Elle contient le nécessaire (compilateur, outils, runtimes et APIs) pour écrire et exécuter des applications et des applets
 - **JME (Java Micro Edition)** : pour les services intégrés. Pour développer des applications capables de fonctionner dans des environnements limités tels que les assistants personnels (PDA), les téléphones mobiles ou les systèmes de navigation embarqués ;
 - **JEE (Java Enterprise Edition)** : pour le traitement côté serveur. Contient plusieurs APIs permettant le développement d'applications d'entreprises (business applications) tel que JDBC pour l'accès aux bases de données, EJB pour développer des composants orientés métiers, Servlet/JSP pour générer des pages HTML dynamiques ...

Caractéristiques de java (voir Doc2)

- Le livre blanc de java cite 11 caractéristiques pour ce langage:
 - Simplicité
 - Orienté objet
 - Compatible avec les réseaux
 - Fiabilité
 - Architecture neutre
 - Portabilité
 - Interprété
 - Performances élevées
 - Multithread
 - Dynamique

Caractéristiques de java (voir Doc2)

Java est portable

- La portabilité est la grande force du langage Java :
 - Un programme réalisé en Java peut fonctionner sur n'importe quelle machine disposant d'un interpréteur Java, que cette machine tourne sous Windows, Unix, MacOS, etc.
 - La compilation du code source Java produit de simples instructions binaires Java (**bytecode**)
 - Ce code est unique et directement opérationnel; il fonctionnera automatiquement sous n'importe quel Système d'Exploitation.
 - Le code généré est interprété et exécuté par un interpréteur Java, appelé couramment **machine virtuelle Java** ...
 - Cette Caractéristique est très importante dans le cadre d'applications exécutées via Internet, des machines très diverses étant connectées.

Caractéristiques de java (voir Doc2)

Java est simple

Java a quasiment la même syntaxe que le C++ , mais il est plus simple: de nombreuses caractéristiques critiques du C++ ont été supprimées:

- Code source simple : Il n'y a plus de préprocesseur :
pas de #define, de typedef, pas de recours aux fichiers headers.
- La gestion de la mémoire et des erreurs est effectuée implicitement en JAVA.
- Les chaînes et les tableaux sont des objets et font partie intégrante du langage.
- Plus de pointeurs, de surcharge des opérateurs, d'héritage multiple...

Mais, ce que Java a gagné en portabilité, il l'a perdu en rapidité: il est beaucoup moins rapide que le C++ ...

Caractéristiques de java (voir Doc2)

Java est robuste et sécurisé

- Le compilateur interdit toute manipulation en mémoire ...
- La gestion de la mémoire n'est pas à la charge du développeur : pas de pointeurs, pas de fonctions d'arguments variables ...
- Compilateurs très strictes : toutes les valeurs doivent être initialisées, contrôle de typage « fort »
- Les limites des tableaux sont vérifiées : Un débordement d'index dans un tableau provoque une erreur (La mémoire n'est pas écrasée)
- Les erreurs à l'exécution sont vérifiées : Le compilateur vérifie que les erreurs sont traitées par le développeur, et si une erreur retournée par une méthode n'est pas traitée, le programme ne compilera pas

Caractéristiques de java (voir Doc2)

Java est un langage objet

- Java reprend des caractéristiques de différents langages à objets, notamment celles de C++
- Toute ligne de code JAVA se trouve obligatoirement dans une méthode à l'intérieur d'une classe
- en Java, tout est objet à l'exception des types primitifs (nombres entiers, nombres à virgule flottante, etc.)
- Nombreuses classes permettant le développement de systèmes répartis
 - Classes d'accès aux ressources d'un serveur Web

Que peut-on faire avec java ?

- On peut réaliser différents types de programmes avec Java :
 - des **applications**, sous forme de fenêtre ou de console:
Elles s'exécutent dans le système d'exploitation à condition d'avoir installé une machine virtuelle ;
 - des **applications web** : applets destinées à fonctionner sur un navigateur , servlets et JSP qui vont s'exécuter du côté serveur ...
 - des **applications** puissantes et efficaces pour appareils mobiles, embarqués, cartes à puce ...
 - et bien d'autres ! J2EE, JMF, J3D pour la 3D...

Exécution d'un programme Java

- Le fichier source d'un programme Java est un simple fichier texte. extension : **.java**
- N'importe quel éditeur de texte peut être utilisé pour éditer un fichier source Java
- Le code source Java est ensuite compilé en "**bytecode**" dans un fichier qui porte le même nom que le fichier source. extension : **.class**
- **Par contre** dans les langages traditionnels: le compilateur crée un fichier binaire directement exécutable par un processeur donné.

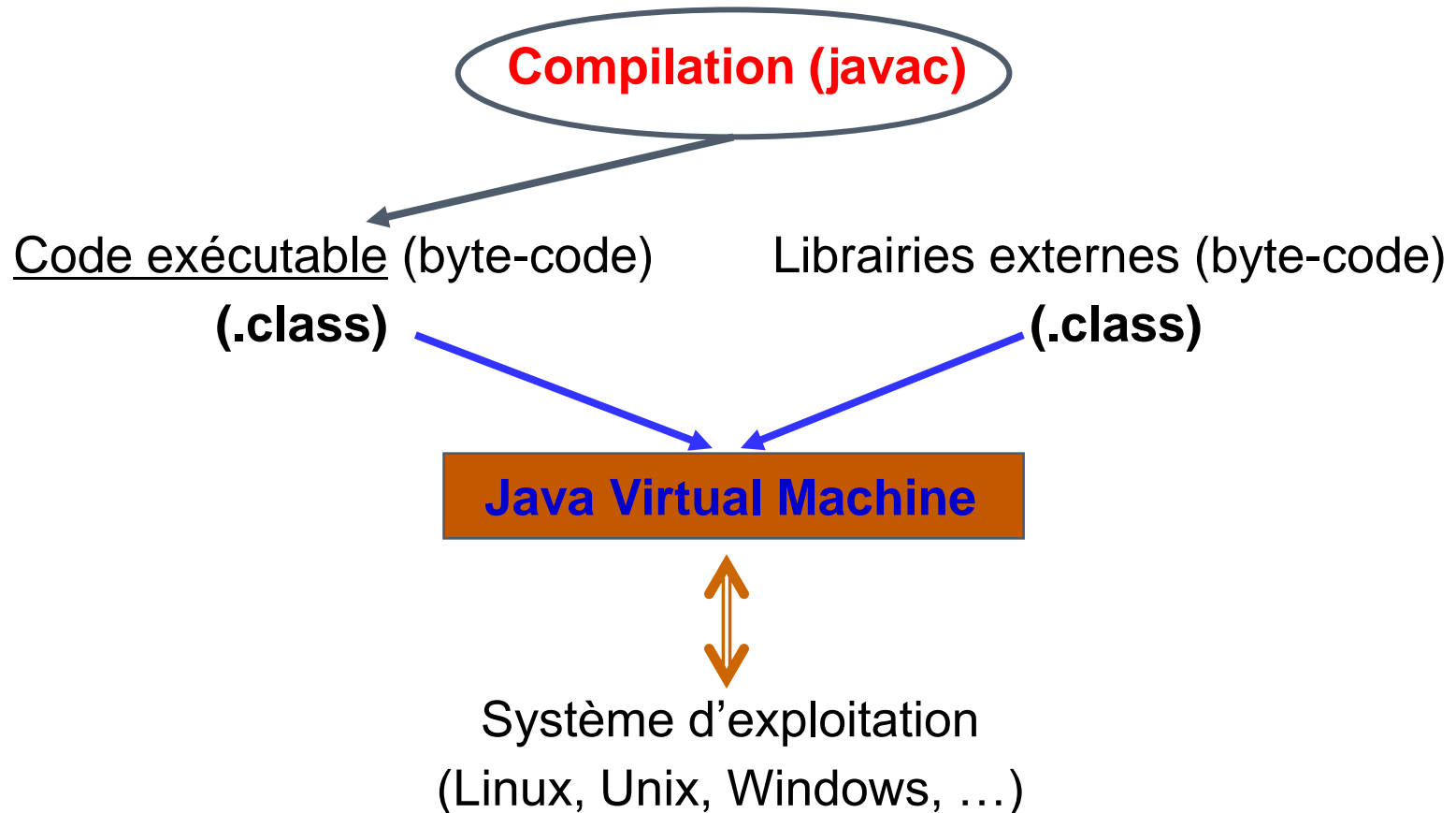
Exécution d'un programme Java

- La compilation d'un programme java se fait soit par l'invocation de la commande **javac** si on travaille en mode console (javac NomFichier.java), soit à partir du menu "Run" si on utilise un IDE.
- C'est cette compilation en bytecode qui fait la portabilité des programmes écrits en Java.
- Une machine virtuelle Java (**JVM** ou **interpréteur Java**) installée sur la plate-forme va permettre l'exécution du code intermédiaire (bytecode).
- La **JVM** est un interpréteur capable d'interpréter le code intermédiaire.
- La **JVM** assure l'utilisabilité d'un programme avec tous les systèmes d'exploitation sur lesquels elle est installée

Exécution d'un programme Java

Un programme Java s'exécute dans une machine virtuelle

Fichier source (.java)



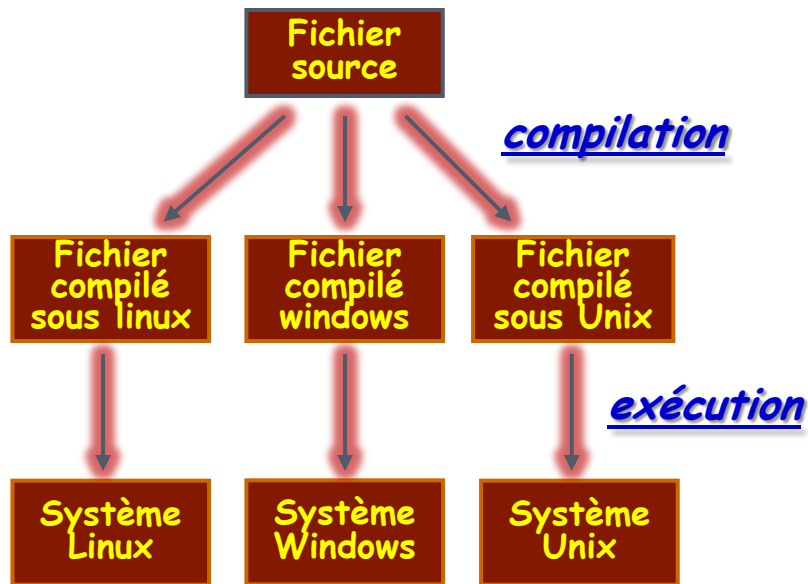
Exécution d'un programme Java

Bytecode

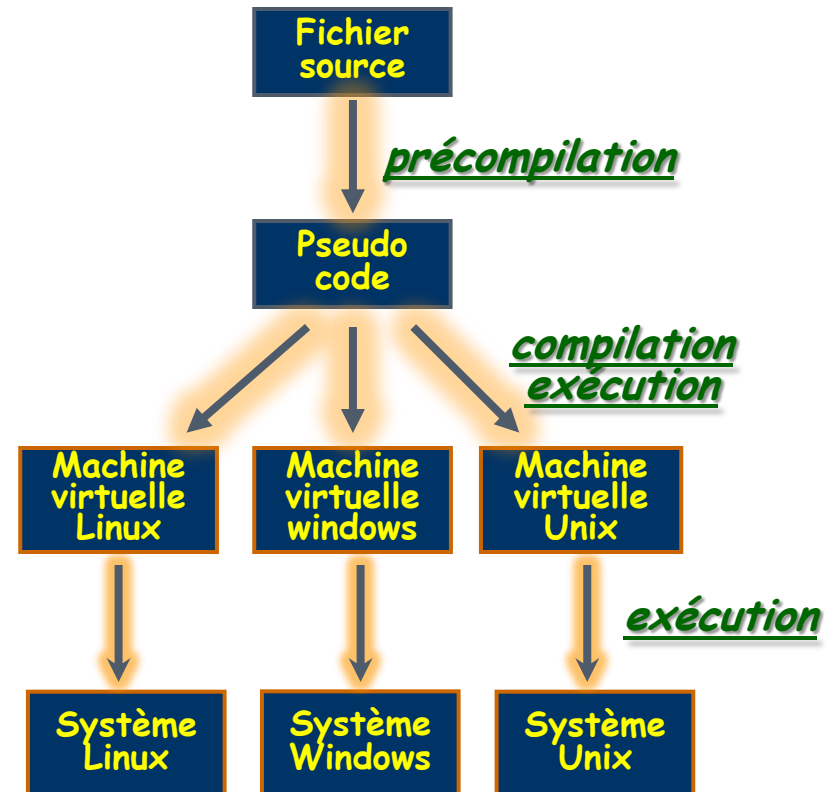
- Le **bytecode Java** est un fichier binaire inintelligible pour notre machine, il est interprétable par la **JVM**.
- C'est un code intermédiaire entre le code Java et le code machine
- C'est un flux d'octets binaire résultat de la compilation d'un code source, qui peut ne pas être écrit en langage java.
- Le bytecode se trouve dans un fichier pré-compilé (extension **.class**)
- Ce *bytecode* est indépendant de la plateforme et peut être exécuté sous de nombreux systèmes d'exploitation par une JVM.
- Le *bytecode* reste le même quelque soit l'environnement où a été développé et pré-compilé le programme Java (=> *un programme Java, codé sous Windows peut être pré-compilé sous Mac et enfin exécuté sous Linux*)

Exécution d'un programme Java

Exécution d'un programme
C/C++ ...



Exécution d'un programme
JAVA



La programmation en langage java nécessite l'installation d'un certain nombre d'outils :

- Un **kit de développement (JDK)** pour pouvoir compiler et interpréter les programmes
 - Un **environnement d'exécution (JRE)** dont le rôle est d'exécuter des programmes Java.
- Un **environnement de développement (IDE)**, pour écrire les codes sources

JDK : Java Development Kit

- Le **JDK** est le kit de développement de base.
C'est un ensemble d'outils et d'API fournis gratuitement par Sun/Oracle, et permettant le développement de programmes avec Java.
C'est l'environnement dans lequel le code Java est compilé et transformé en bytecode puis interprété par la machine virtuelle.
- Le **JDK** fourni :
 - un compilateur java
 - une machine virtuelle java « jvm »
 - un ensemble de bibliothèques d'outils pour faciliter la programmation.

JDK : Java Development Kit

- Le **JDK** est constitué de plusieurs outils dont :
 - **javac**: le compilateur, qui convertit le code source en bytecode (contenu dans un fichier **.class**)
 - **java**: un interpréteur d'applications (machine virtuelle); il interprète le bytecode généré par le compilateur Java (javac)
 - **applet viewer**: un interpréteur d'applets
 - **jar**: utilitaire permettant de compresser les classes Java afin de réduire leur taille et de rendre leur téléchargement plus rapide
 - **javap**: un décompilateur, pour revenir du bytecode au code source
 - **jdb**: le débogueur java pour localiser les erreurs ...

JDK : Java Development Kit

➤ Où se procurer le JDK

Le JDK est disponible gratuitement en téléchargement sur le site de *Sun Microsystems* :

<http://java.sun.com>

<http://javasoft.com>

➤ Il existe plusieurs versions du JDK pour chaque plateforme (Unix, Solaris, Windows, ...)

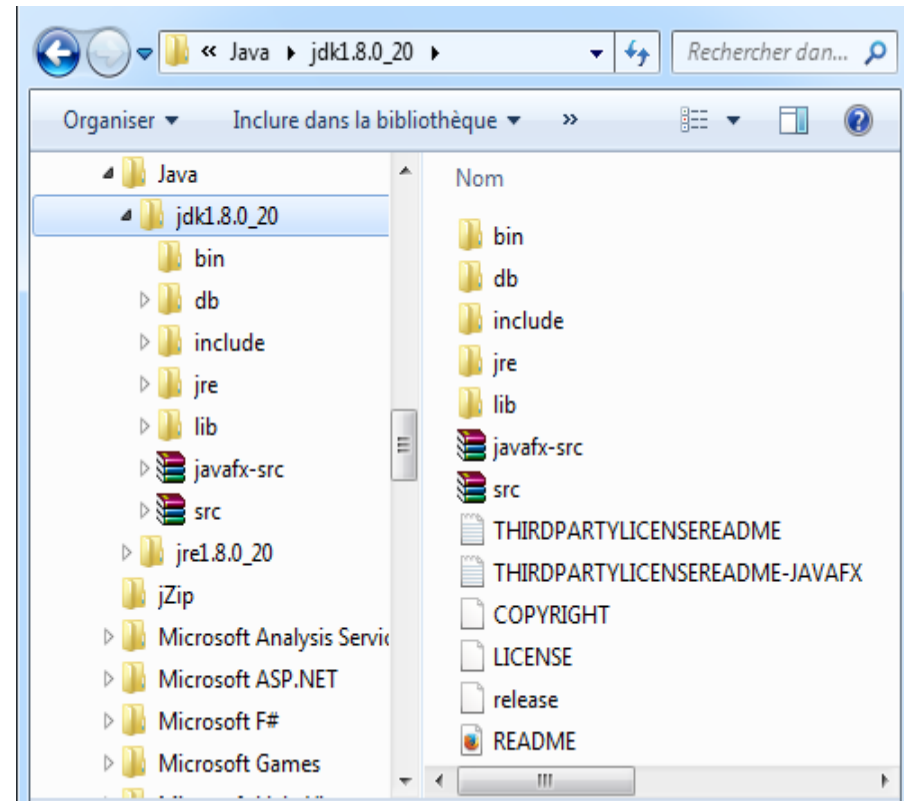
➤ Version utilisée dans ce cours : jdk1.8.0_20 pour windows

www.oracle.com/technetwork/java/javase/overview/install-jdk6-22nb691-177131.html

JDK : Java Development Kit

➤ Arborescence des répertoires du JDK

- Fichier src.zip, contenant les fichiers sources de la bibliothèque du JDK
- Répertoire 'bin', contenant tous les outils du JDK
- Le fichier **README.html**



JDK : Java Development Kit

➤ Arborescence des répertoires du JDK

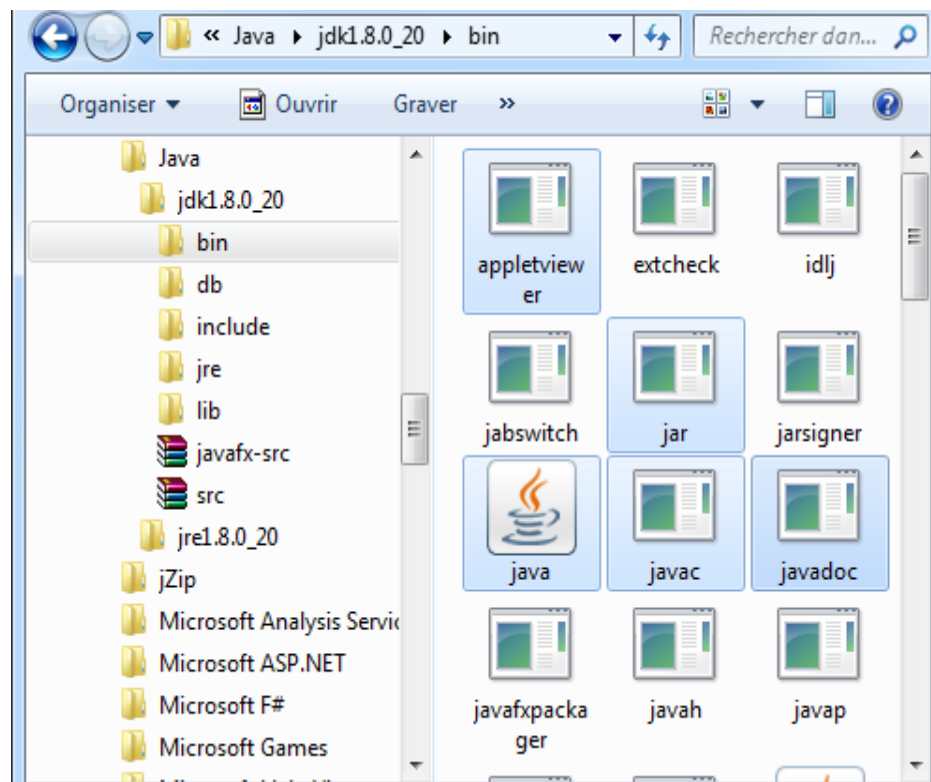
Structure du répertoire	Description
jdk	Le nom peut être différent, par exemple jdk8.0
bin	Compilateur et outils
db	
include	Fichiers pour les méthodes natives
jre	Fichiers d'environnement d'exécution de Java
lib	Fichiers de bibliothèque
src	Source de bibliothèque (après décompression de src.zip)
javafx-src	Source de bibliothèque de JavaFX JavaFX est un ensemble de technologies destinées à faciliter l'écriture d'applications riches graphiquement pour une grande variété de cibles (ordinateur, téléphones portables, etc.)

JDK : Java Development Kit

➤ Contenu du dossier bin

- Les outils les plus importants sont mis en évidence

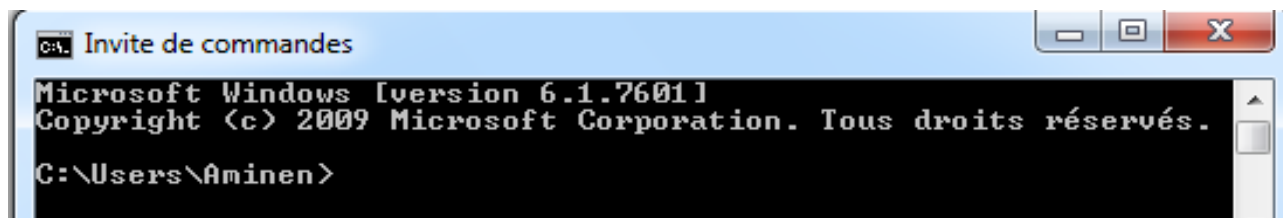
- Le compilateur : **javac**
- La machine virtuelle : **java**
- L'outil de génération de documentation : **Javadoc**
- L'outil de création de 'livrables' pour les clients : **jar**



JDK : Java Development Kit

➤ Test de l'installation

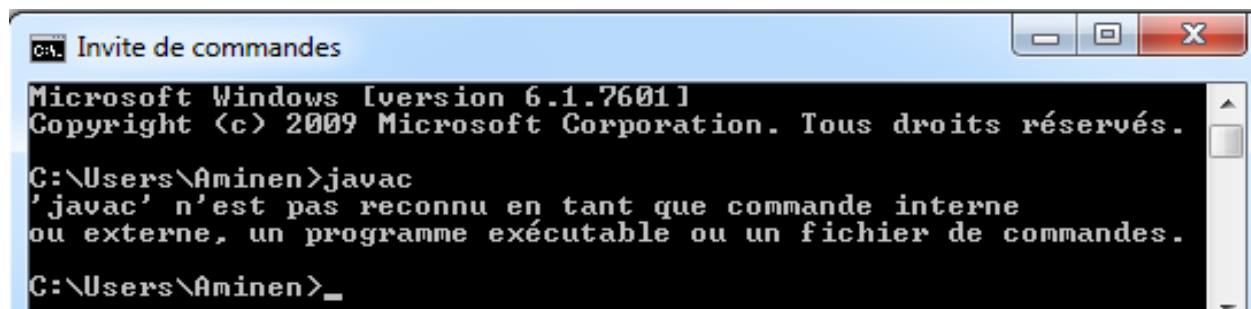
- Lancez une fenêtre de commande MS-DOS : Démarrer/Programmes/Accessoires/Invite de commandes.



```
Microsoft Windows [version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\Aminin>
```

- Taper la commande javac



```
Microsoft Windows [version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\Aminin>javac
'javac' n'est pas reconnu en tant que commande interne
ou externe, un programme exécutable ou un fichier de commandes.

C:\Users\Aminin>_
```

le système ne reconnaît pas cette commande

JDK : Java Development Kit

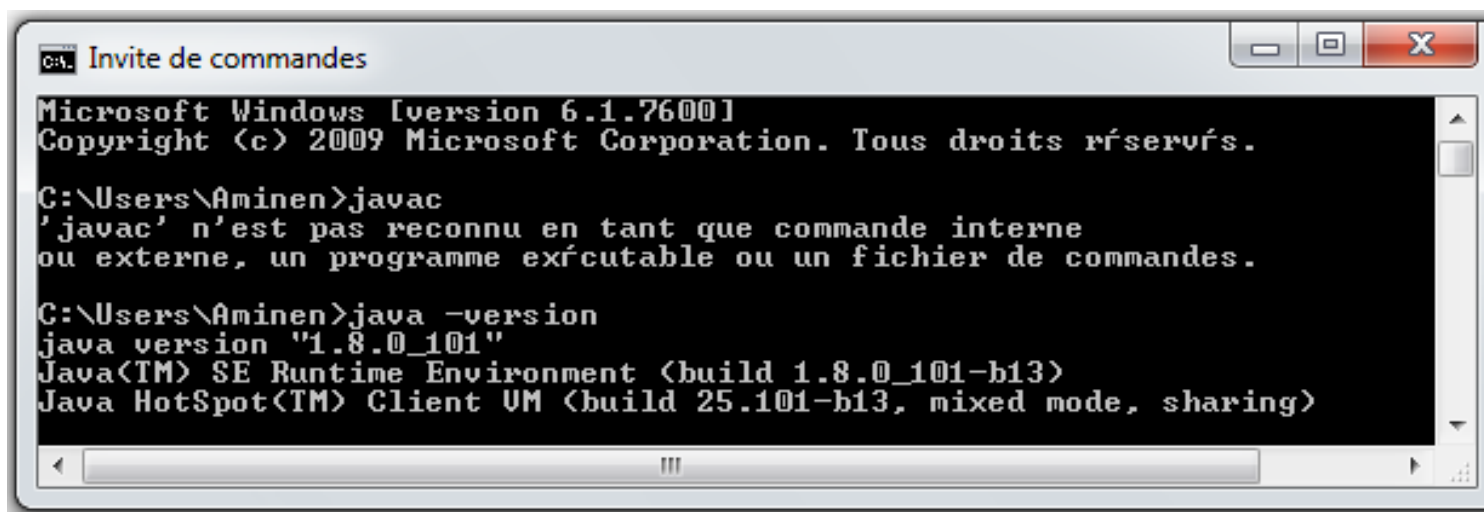
➤ Configurer le chemin d'exécution

- Il faut ajouter le répertoire jdk/bin au chemin d'exécution, la liste des répertoires que traverse le système d'exploitation pour localiser les fichiers exécutables.
- Aller dans: Panneau de configuration\Systeme et sécurité\Systeme\Paramètres système avancés\
- Choisir: Variables d'environnement
- Dans variables système cliquer sur la variable **Path** puis sur **modifier** (si elle n'existe pas cliquer sur **Nouvelle**)
- et ajouter : C:\Program Files\Java\jdk1.8.0_20\bin; *le reste*

JDK : Java Development Kit

➤ Configurer le chemin d'exécution

- Sauvegarder la configuration et redémarrer le PC.
- Ouvrir une NOUVELLE fenêtre MS-DOS, et lancer respectivement les commandes : `java -version`, `java` et `javac`
- Pour `java -version`, on obtient :



```
Microsoft Windows [version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\Aminin>javac
'javac' n'est pas reconnu en tant que commande interne
ou externe, un programme exécutable ou un fichier de commandes.

C:\Users\Aminin>java -version
java version "1.8.0_101"
Java(TM) SE Runtime Environment (build 1.8.0_101-b13)
Java HotSpot(TM) Client VM (build 25.101-b13, mixed mode, sharing)
```

JRE : Environnement d'exécution Java

- **JRE (Java Runtime Environment)** ("environnement d'exécution Java"): ensemble d'outils nécessaires à l'exécution d'applets et d'applications créés à l'aide du langage Java, sur toutes les plates-formes supportées.
- Un **JRE** ne contient pas le kit d'outils pour développer des applications java.
- **JRE** est constitué de:
 - **JVM** (Java Virtual Machine - Machine Virtuelle Java), le logiciel qui interprète le code Java compilé (bytecode) et le convertit en code natif.
 - Une bibliothèque standard à partir de laquelle doivent être développés tous les programmes en Java.

JRE : Environnement d'exécution Java

Différence entre JRE et JDK

JRE (Java Runtime environment)	JDK (Java Development Kit)
Il s'agit d'une implémentation de la machine virtuelle Java qui exécute des programmes Java.	Il s'agit d'un bundle de logiciels qu'on peut utiliser pour développer des applications Java.
Java Runtime Environment est un plug-in requis pour l' exécution de programmes Java.	Le kit de développement Java (JDK) est requis pour le développement d'applications Java.
Le JRE est moins volumineux que le JDK et requiert donc moins d'espace disque.	Le JDK requiert davantage d'espace disque car il contient le JRE ainsi que divers outils de développement.
Il contient le JVM, les bibliothèques de base et d'autres composants supplémentaires pour l'exécution d'applications et d'applets écrits dans Java.	Il contient le JRE, un jeu de classes API, un compilateur Java, Web Start et d'autres fichiers requis pour l'écriture d'applets et d'applications Java.

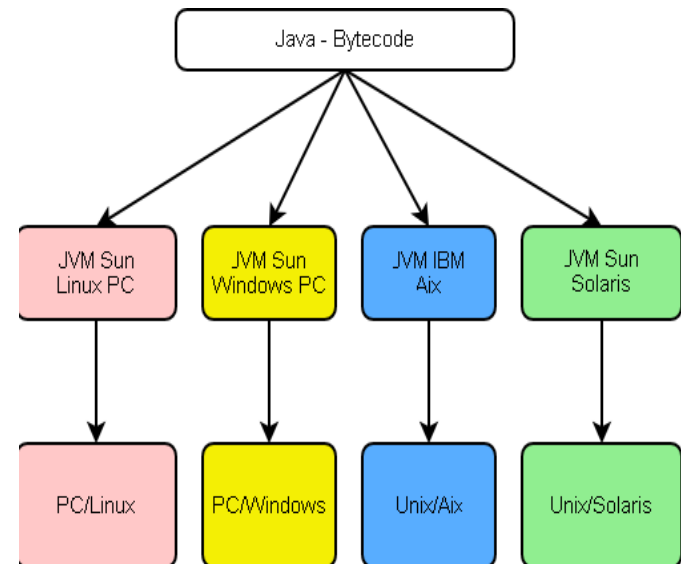
JRE : Environnement d'exécution Java

JVM : Machine Virtuelle Java

- La **JVM** ou **machine virtuelle java** est un interpréteur java, c'est un logiciel qui permet d'interpréter le bytecode Java.
- Ce programme est spécifique à chaque plate-forme (couple machine/SE) et permet aux applications Java compilées en bytecode de produire les mêmes résultats sur toute plate-forme pourvue de la JVM adéquate.

différentes JVM
une JVM par plate-forme

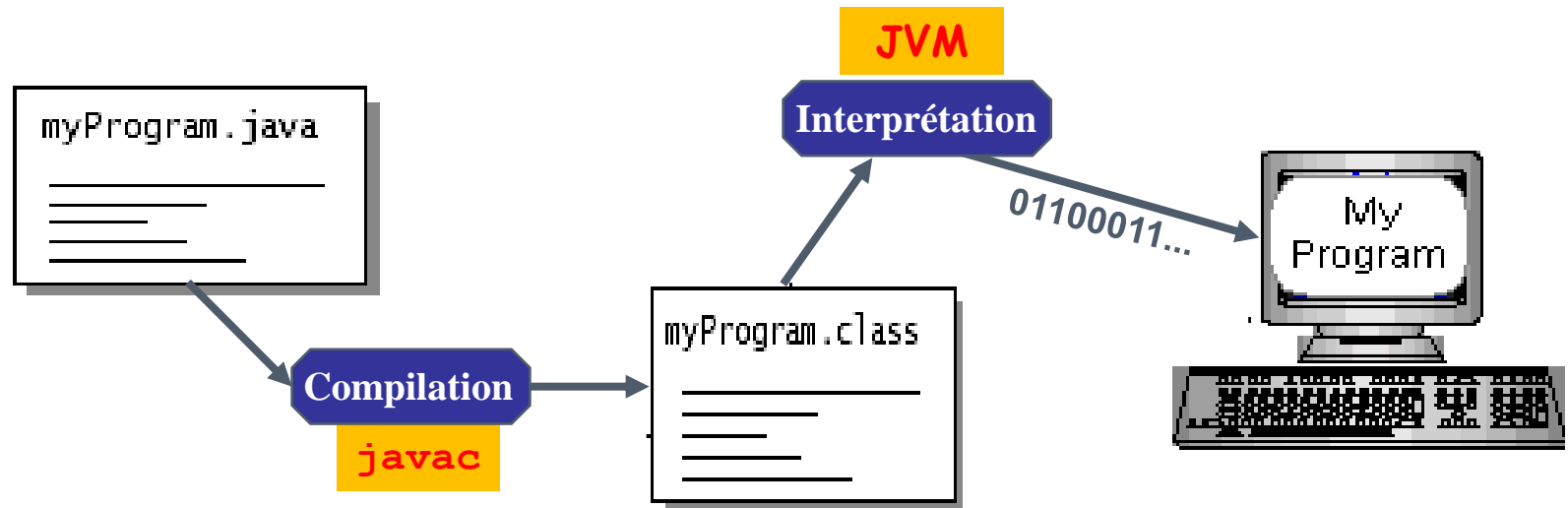
différentes plateformes



JRE : Environnement d'exécution Java

JVM : Machine Virtuelle Java

- La compilation génère un fichier (.class) en bytecode
- Le bytecode est interprété par un interpréteur Java JVM



Environnement de développement (IDE)

- Avec le JDK tout se fait par l'entrée de commandes dans une fenêtre Invite de commandes. Il est plus simple d'utiliser un environnement de développement professionnel ou IDE.
- Un **IDE** (Integrated **D**evelopment **E**nvironment) est un logiciel qui va permettre de développer les applications ou les applets, et de les compiler. Il va permettre de traduire les programmes Java en langage compilé.

Environnement de développement (IDE)

- Les **IDE** offrent des fonctions avancées :
 - Éditeur évolué (couleur, autocorrection, présentation...)
 - Assistance syntaxique
 - Générateur de code
 - Débugage
 - Environnement d'exécution ...

- **exemples d'IDE**

- Eclipse
- NetBeans (Sun Microsystems)
- JBuilder (Borland)
- Visual J++ (Microsoft) ...

On peut aussi tout simplement utiliser Notepad ou Bloc Notes

Récapitulatif ...

- Pour développer et exécuter des programmes java, on peut utiliser un **JDK** (compilateur+jvm) ou un **IDE** avec un **JRE**.
- Avec l'**IDE** on écrit le code source java: extension **.java**
- les fichiers pré-compilés correspondant aux codes sources Java ont l'extension **.class**
- la JVM, coeur de Java, fait fonctionner les programmes Java, pré-compilés en bytecode
- le bytecode est un code intermédiaire entre le programme et votre machine
- un programme Java, codé sous un SE peut être pré-compilé sous un autre et enfin exécuté sous un autre
- notre machine NE PEUT PAS interpréter du bytecode

Notion de Garbage Collector

- En programmation java, l'allocation de la mémoire pour un objet est automatique à sa création.
- Si l'objet est détruit, Java récupère automatiquement la mémoire inutilisée grâce au garbage collector.
- Le garbage collector restitue les zones de mémoire laissées libres suite à la destruction des objets
- Le garbage collector gère la désallocation de la mémoire :
 - le programmeur alloue la mémoire dont il a besoin
 - le programmeur ne désalloue pas la mémoire :
 - plus de risque de fuite mémoire
 - plus de risque d'utiliser un pointeur désalloué.

Notion de Garbage Collector

- A l'inverse de nombreux langages orientés objet tel que le C++ ou Delphi, le programmeur Java n'a pas à se préoccuper de la destruction des objets qu'il instancie. Ceux-ci sont détruits et leur emplacement mémoire est récupéré par le ramasse miette de la machine virtuelle dès qu'il n'y a plus de référence sur l'objet.
- La machine virtuelle garantit que toutes les ressources Java sont correctement libérées