JAVA PLATE-FORME

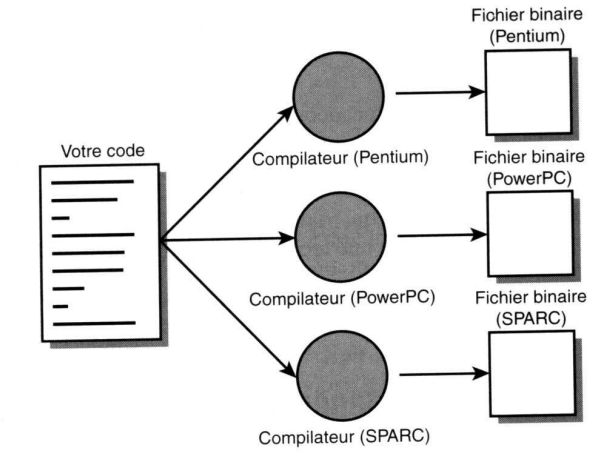
Apparition du langage

Vers la fin de 1995, le langage de programmation Java surgit sur la grande scène d'Internet et obtient immédiatement un énorme succès. La prétention de Java est de constituer la colle universelle capable de connecter les utilisateurs aux informations, que celles-ci proviennent de serveurs Web de bases de données, de fournisseurs d'informations ou de toute autre source imaginable. Et Java se trouve en bonne position pour accomplir ce pari. Il s'agit d'un langage de conception très performant qui a été adopté par la majorité des fournisseurs. Ses caractéristiques intégrées de sécurité offrent un sentiment de confiance au programmeurs comme aux utilisateurs des applications. De plus, Java incorpore des fonctionnalités qui facilitent grandement certaines tâches de programmation avancées comme la gestion des réseaux, la connectivité des bases de données ou le développement d'applications multitâches.

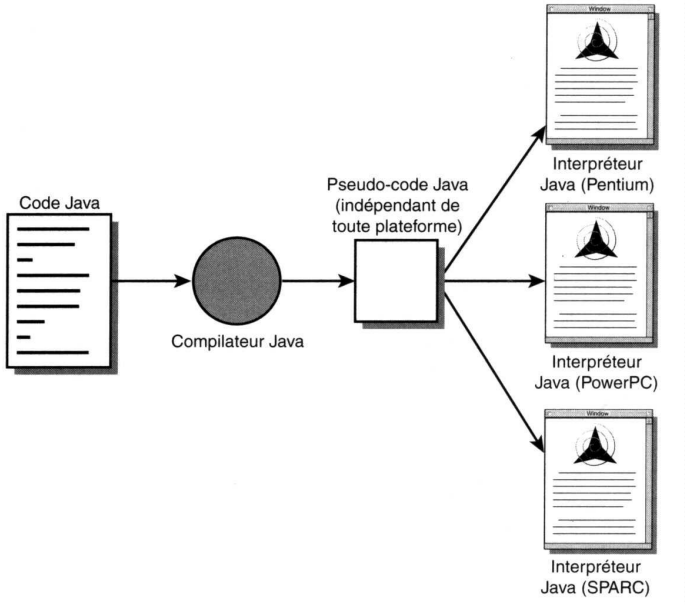
La réputation de Java en tant que langage informatique est exagérée: Java est assurément un *bon* langage de programmation. Il s'agit, sans aucun doute, de Fun des meilleurs disponibles pour un programmeur sérieux. Java aurait, *potentiellement*, *pu* être un grand langage de programmation, mais il est probablement trop tard pour cela. Lorsqu'un langage commence à être exploité se pose le problème de la compatibilité avec le code existant. De plus, même lorsque des modifications sont possibles sans révolutionner le code existant, il est très difficile pour les créateurs d'un langage qui a suscité autant d'intérêt de reconnaître qu'un élément x pourrait être plus mauvais ou meilleur qu'un élément y. Même si nous espérons une amélioration de Java avec le temps, sa structure de demain restera, à la base, très proche de celle d'aujourd'hui.

Les avantages de Java

L'un des avantages évidents de ce langage est une bibliothèque d'exécution qui se veut indépendante de la plate-forme: en théorie, il vous est possible d'utiliser le même code pour Windows 95/98/NT, Solaris UNIX Macintosh, etc. Cette propriété est indispensable pour une programmation sur Internet (cependant, par rapport à la disponibilité sur Windows et Solaris les implémentations sur d'autres plates-formes ont toujours un léger décalage).



1) Architecture classique avec un bytecode différent pour chaque processeur .



2) Architecture Java, le bytecode passe par l'intermédiaire d'un interpréteur.

Un autre avantage de ce langage de programmation réside dans le fait que la syntaxe de Java est analogue à celle de C++ ce qui le rend économique et professionnel.

Le fait de créer une autre version d'un langage C++ n'est cependant pas suffisant. Le point clé est le suivant : il est beaucoup plus facile d'obtenir du *code sans erreur à l'aide de java qu'avec* C++
Pourquoi ? Les concepteurs de Java ont beaucoup réfléchi à la raison pour laquelle le code C++ contenait autant d'erreurs. Cette réflexion les a amenés à ajouter dans Java des fonctions destinées à éliminer la possibilité de créer du code contenant les types d'erreurs les plus courants (selon certaines estimations, le code C++ contient au moins une erreur toutes les cinquante lignes).

• Les concepteurs de java ont supprimé l'allocation et la libération de mémoire manuelles. La mémoire dans java est allouée et libérée automatiquement. Vous n'avez *jamais* à vous préoccuper de pertes de mémoire.
• Ils ont éliminé l'arithmétique des pointeurs introduisant du même coup une vraie gestion de tableau. La notion de référence sur une zone mémoire remplace avantageusement celle de " pointeur", car elle supprime la possibilité d'écraser toute zone mémoire à cause d'un compteur erroné.
• Ils ont éliminé toute possibilité de confusion entre une affectation et un test d'égalité ans une instruction conditionnelle. Une instruction `if (ntries - 3)` ne pourra pas franchir l'étape de la compilation
• Ils ont supprimé l'héritage multiple en le remplaçant par une nouvelle notion d'interface dérivée d'Objective C. Les interfaces vous offrent tout ce que vous pouvez obtenir à partir de l'héritage multiple, sans la complexité de la gestion de hiérarchie d'héritage multiple.

Caractéristiques

Les créateurs de Java ont écrit un livre blanc qui présent les caractéristiques fondamentales de Java. Ce livre est articulé autour des 11 termes suivants :

- ***Distribué***

Java possède une importante bibliothèque de routines permettant de gérer les protocoles TCP/IP tels que HTTP et FTP. Les applications Java peuvent charger et accéder à des sur Internet via des URL *avec la même facilité qu'elles accèdent à un fichier local sur le système.*
« Nous avons trouvé que les fonctionnalités réseau de Java sont à la fois fiables et d'utilisation aisée. Toute personne ayant essayé de faire de la programmation pour Internet avec un autre langage se réjouira de la simplicité de Java lorsqu'il s'agit de mettre en oeuvre des tâches lourdes, comme l'ouverture d'une connexion avec un socket. De plus, Java rend plus facile l'élaboration des scripts CGI *(Common Gateway Interface)*, et un mécanisme élégant, nommé *servlet*, augmente considérablement l'efficacité du traitement côté serveur, assuré par Java. De nombreux serveurs Web, parmi les plus courants, supportent les servlets. Le mécanisme d'invocation de méthode à distance (RMI) autorise la communication entre objets distribués. »

- ***Fiabilité***

Java a été conçu pour que les programmes qui l'utilisent soient fiables sous différents aspects. Sa conception encourage le programmeur à traquer préventivement les éventuels problèmes, à lancer des vérifications dynamiques en cours d'exécution et à éliminer les situations génératrices d'erreurs... La seule et unique grosse différence entre C++ et Java réside dans le fait que ce dernier intègre un modèle de pointeur qui écarte les risques d'écrasement de la mémoire et d'endommagement des données.

- ***Orienté objet***

Pour rester simples, disons que la conception orientée objet est une technique de programmation qui se concentre sur les données (les objets) et sur les interfaces avec ces objets. Pour faire une analogie avec la menuiserie, on pourrait dire qu'un menuisier "orienté objet " s'intéresse essentiellement à la chaise l'objet qu'il fabrique et non à sa conception (le "comment"). Par opposition, le menuisier "non orienté objet " penserait d'abord au "comment "...

- ***Simple***

Nous avoirs voulu créer un système qui puisse être programmé simplement sans nécessiter un apprentissage ésotérique, et qui tire parti de l'expérience standard actuelle. En conséquence, même si nous pensions que C++ ne convenait pas, Java a été conçu de façon relativement proche de ce langage dans le dessein de faciliter la compréhension du système. De nombreuses fonctions compliquées, mal comprises, rarement utilisées de C++, qui nous semblaient par expérience apporter plus d'inconvénients que d'avantages, ont été supprimées de Java.

- **Sécurité**

Java a été conçu pour être exploité dans des environnements serveur et distribués. Dans ce but, la sécurité n'a pas été négligée. Java permet le construction de systèmes inaltérables et sans virus.

- **Architecture neutre**

Le compilateur génère un format de fichier objet dont l'architecture est neutre – le code compilé est exécutable sur de nombreux processeurs, à partir du moment où le système d'exécution de Java est présent. Pour ce faire, le compilateur Java génère des instructions en bytecode qui n'ont de lien avec aucune architecture particulière. Au contraire, ces instructions ont été conçues pour être à la fois faciles à interpréter et faciles à traduire en code natif.

- **Portable**

A la différence du C/C++, on ne trouve pas les aspects de dépendance de la mise ne œuvre dans la spécification. Les tailles des types de données primaires sont spécifiées, ainsi que le comportement arithmétique qui leur est applicable.

- **Interprété**

L'interpréteur Java peut exécuter les bytecode directement sur n'importe quelle machine sur laquelle il a été porté. Dans la mesure où la liaison est un processus plus incrémentiel et léger, le processus de développement peut se révéler plus rapide et exploratoire.

- **Performances élevées**

En général, les performances des bytecodes interprétés sont tout à fait suffisantes, il existe toutefois des situations dans lesquelles des performances plus élevées sont nécessaires. Les bytecodes peuvent être traduits à la volée en code machine pour l'unité centrale destinée à accueillir l'application.

- **Multithread**

Les avantages du multithread sont une meilleure interréactivité et un meilleur comportement en temps réel.