LANGAGE ORIENTE OBJET

Grégory Bourguin

gregory.bourguin@univ-littoral.fr

Historique

- Simula 67 a été conçu par une équipe scandinave et a été publié en 1967.

- Au début des années 70, Alan Kay conçoit au PARC (Rank Xerox) le langage SmallTalk, qui est encore aujourd'hui LA référence dans les langages orientés objet.

Historique

- A la fin des années 70 et au début des années 80, on assiste à la naissance de nombreuses extensions objet d'anciens langages : Object Pascal, Objective C, C++, CLOS, ADA...
- Au milieu des années 90, Sun publie Java. Ses qualités intrinsèques et le fait qu'il soit particulièrement adapté à Internet en font immédiatement un standard.
- Une énorme masse de documentation peut être trouvée sur Internet

Qu'est ce que Java ?

- Un langage de Programmation Orienté Objet (POO)
- Une architecture de Virtual Machine
- Un ensemble d'API variées
- Un ensemble d'outils (le JDK)

Qu'est ce que Java ?

- Java != JavaScript (c'est un langage généraliste, type C++)
- Java != C++ (c'est un langage purement objet, de plus haut niveau, plus proche de SmallTalk)

Portable

- Le compilateur Java génère du byte code.
- La Java Virtual Machine (JVM) est présente sous Unix, Windows, Mac, ...
- Le langage a une sémantique très précise.
- Java supporte un code source écrit en Unicode.
- Java est accompagné d'une librairie standard.

Robuste

- A l'origine, c'est un langage pour les applications embarquées.
- Gestion de la mémoire par un garbage collector.
- Mécanisme d'exception.
- Seules les conversions sûres sont automatiques.
- Contrôle des cast à l'exécution.

Les outils

- **Environnements de développement** :
  - JDK (compilateur, interpréteur, ...)
  - IDE : Eclipse, NetBeans, IntelliJ…

Les outils

- **javac** : compilateur de sources java
- **java** : interpréteur de *byte code*
- **appletviewer** : interpréteur d'applet
- **javadoc** : générateur de documentation (HTML, MIF)
- **javah** : générateur de *header* pour l'appel de méthodes natives
- **javap** : désassembleur de *byte code*
- **jdb** : debugger
- **javakey** : générateur de clés pour la signature de code
- **rmic** : compilateur de *stubs* RMI
- **rmiregistry** : "*Object Request Broker*" RMI

APIs standards

- **java.lang** : types de bases, ...
- **java.util** : Hashtable, Vector, Stack, Date, …
- **java.io** : accès aux i/o par flux
- **java.net** : socket (UDP, TCP, multicast), URL, ...
- **java.rmi** : *Remote Method Invocation*
- **java.lang.reflect** : introspection sur les classes et les objets
- **java.beans** : composants logiciels
- **java.sql** (JDBC) : accès homogène aux bases de données
- **java.security** : signature, cryptographie, authentification
- …