

Introduction à Java

Philippe Genoud

Philippe.Genoud@imag.fr

~~<http://www.inrialpes.fr/helix/people/genoud/ENSJAVA>~~

<http://membres-liglab.imag.fr/genoud/ENSJAVA/M2CCI/>

JAVA c'est quoi ?

- **Une technologie développée par SUN Microsystems™ lancée en 1995**
 - Un langage de programmation
 - Une plateforme , environnement logiciel dans lequel les programmes java s'exécutent.
- **Présente dans de très nombreux domaines d'application : des serveurs d'applications aux téléphone portables et cartes à puces (JME)**
 - Quelques chiffres (Java One 2005):
 - 4,5 millions de développeurs Java
 - 700 millions de PC équipés d'une JVM
 - 708 millions de téléphones Java
 - 600 millions de périphériques mobiles
 - 1 milliard de Java Cards

Le Langage Java

- Dans un des premiers papiers* sur le langage JAVA, SUN le décrit comme suit :

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

* *White Paper : The Java Language Environment* - James Gosling, Henry McGilton - May 1996
<http://java.sun.com/docs/white/langenv/>

Le Langage Java

Mon premier programme Java (pas très objet...)

Le code de la classe doit être enregistré dans un fichier de même nom (casse comprise) que la classe



HelloWorld.java

public class HelloWorld {

public static void main(String[] args) {

int i = 0;

while (i < 5) {

System.out.println("Hello World !");

i++;

}

}

}

Le point d'entrée pour l'exécution est la méthode `main()`

① Tout code java doit être défini à l'intérieur d'une **classe**

② La description de la classe est effectuée à l'intérieur d'un bloc `{ }`

Compilation :

`javac HelloWorld.java`



HelloWorld.java

javac

HelloWorld.class

Exécution :

`java HelloWorld`



java

Hello World !
Hello World !
Hello World !
Hello World !
Hello World !

Le langage Java

Un langage compilé / interprété

- Compilation d'un programme JAVA : génération de byte-code

```
public class Test {  
  
    public static void main(String[] args)  
    {  
        for (int i = 0; i < 10; i++)  
            System.out.println("Hello " + i);  
    }  
}
```

Test.java
code source

javac

```
01100001  
11100000  
11111010  
11101...  
...
```

Test.class
byte-code

javap -c Test
Désassemble Test.class

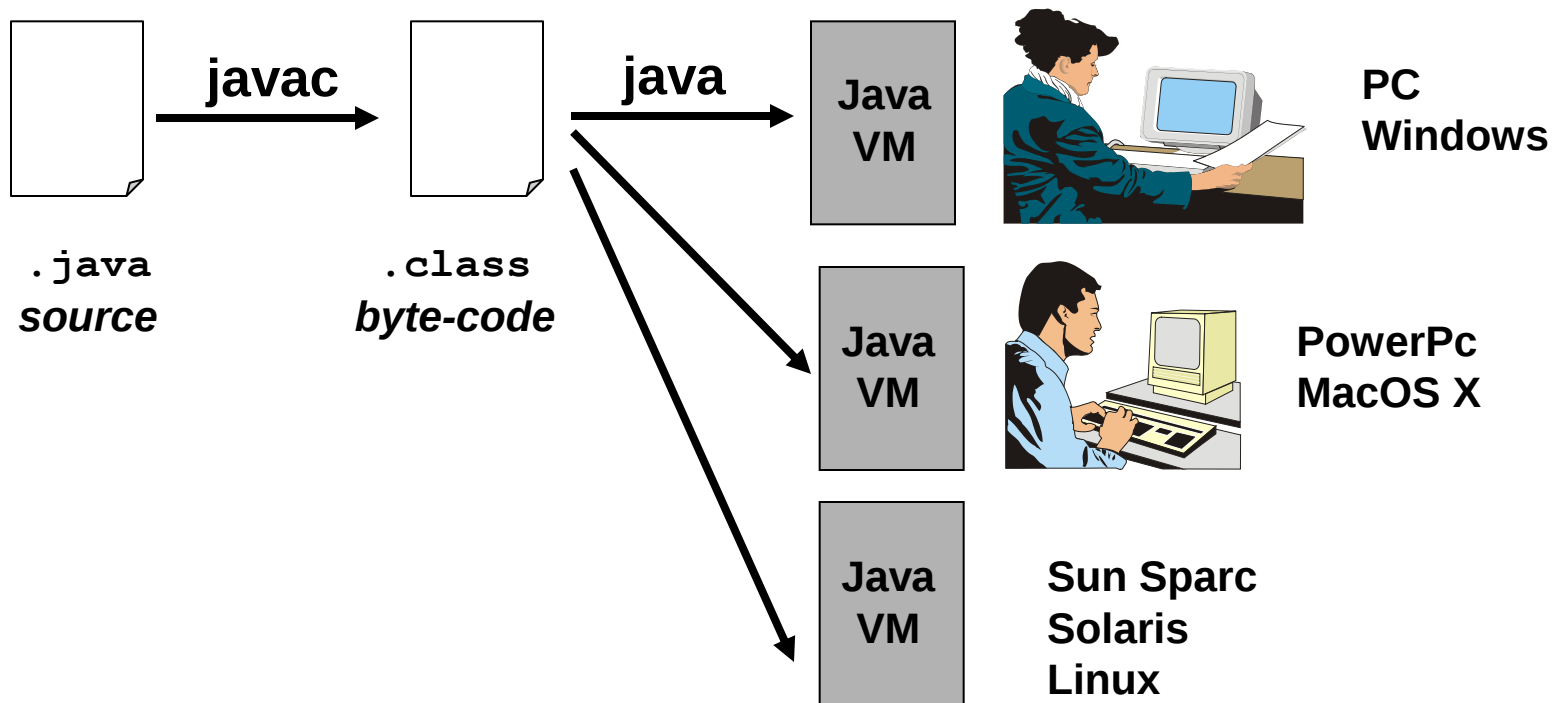
```
0  iconst_0  
1  istore_1  
2  goto 30  
5  getstatic #10 <Field java.lang.System.out>  
8  new #5 <Class java.lang.String>  
11 dup  
12 ldc #1 <String "Hello ">  
...  
27 iinc 1 1  
30 iload_1  
31 bipush 10  
33 if_icmplt 5  
36 return
```

- Le byte-code est :
 - proche d'un langage machine
 - indépendant de la plateforme d'exécution (matériel + OS)

La machine virtuelle Java

Exécution d'un programme Java compilé

- **byte-code assure la portabilité des programmes Java**
 - langage d'une Machine Virtuelle
 - à l'exécution un interpréteur simule cette machine virtuelle



La machine virtuelle java

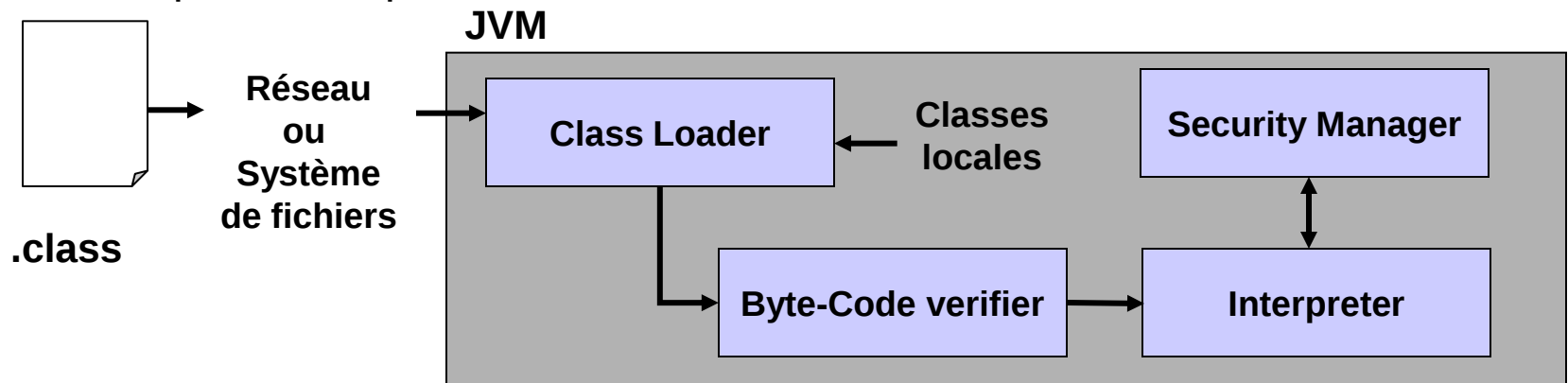
Principes de fonctionnement

- **Chargement**

- chargement sélectif et dynamique des classes
- vérification statique du code (tentatives de modification de la machine virtuelle, ...)

- **Protection lors de l'exécution**

- Le "security manager" possède un droit de veto (accès "sauvages" au système de fichiers, ...)
- Cette protection peut-être modulée

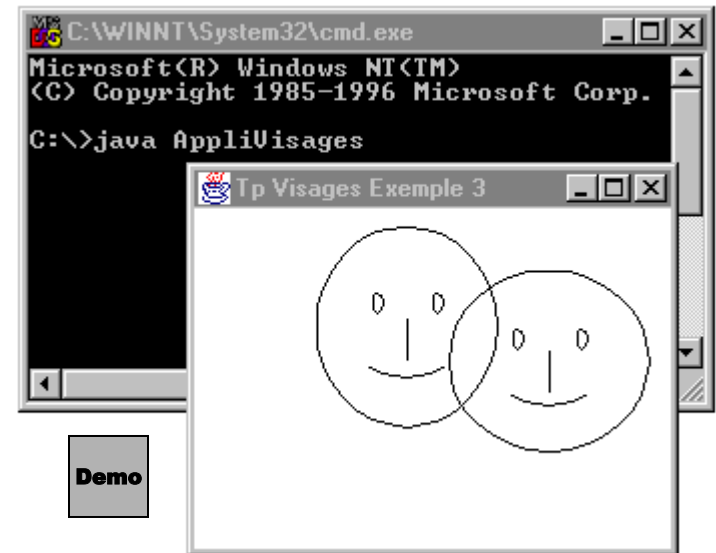


Performances

- **Exécution d'un programme Java**
 - le code Java est compact,
 - le chargement des classes nécessaires est sélectif et dynamique,
 - ... mais Java est interprété
- **Palier aux lenteurs de l'interprétation**
 - utilisation d'un JIT (compilateur « Just-in-Time »)
 - compilation à la volée du byte-code
 - réutilisation du code déjà compilé
 - intégration du JIT HotSpot dans JVM depuis version 1.3
 - *Performance of Java versus C++ J.P.Lewis and Ulrich Neumann*
(<http://www.idiom.com/%7Ezilla/Computer/javaCbenchmark.html>)

Types de programmes Java

- **Les Applications indépendantes**
 - Programmes autonomes (stand-alone)



- **Les Applets**
 - Programmes exécutées dans l'environnement d'un navigateur Web et chargés au travers de pages HTML

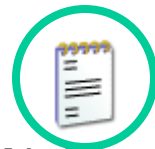
Seuls diffèrent les contextes d'invocation et d'exécution

- Les droits des applets et des applications ne sont pas les mêmes

Types de programmes Java

Application indépendante

Application est définie par un ensemble de classes dont **une** jouera le rôle de **classe principale**



AppliVisage.java
La classe "principale"



VisageRond.java
Dessine un visage



Dessin.java
Gère une liste de visages

javac AppliVisage.java

La compilation de la classe principale entraîne la compilation de toutes les classes utilisées

javac $\simeq$ make



AppliVisage.class



VisageRond.class

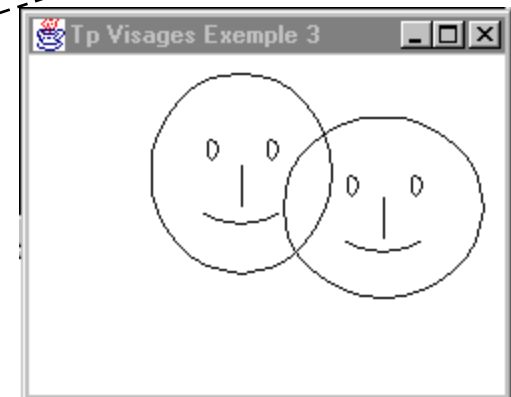


Dessin.class

Pour exécuter l'application on indique à l'interpréteur **java** le nom de la classe principale

java charge les classes nécessaires au fur et à mesure de l'exécution

java AppliVisage



Types de programmes Java

Application indépendante

- **Application doit posséder une classe principale**

- classe possédant une méthode de signature

Tableau de chaînes de caractères
(équivalent à `argc, argv` du C)

```
public static void main(String[] args)
```

- **Cette méthode sert de point d'entrée pour l'exécution**

- l'exécution de l'application démarre par l'interprétation de cette méthode

Exécute le code défini dans la méthode
main contenue dans le fichier
`AppliVisage1.class`

```
ex: java AppliVisage1
```

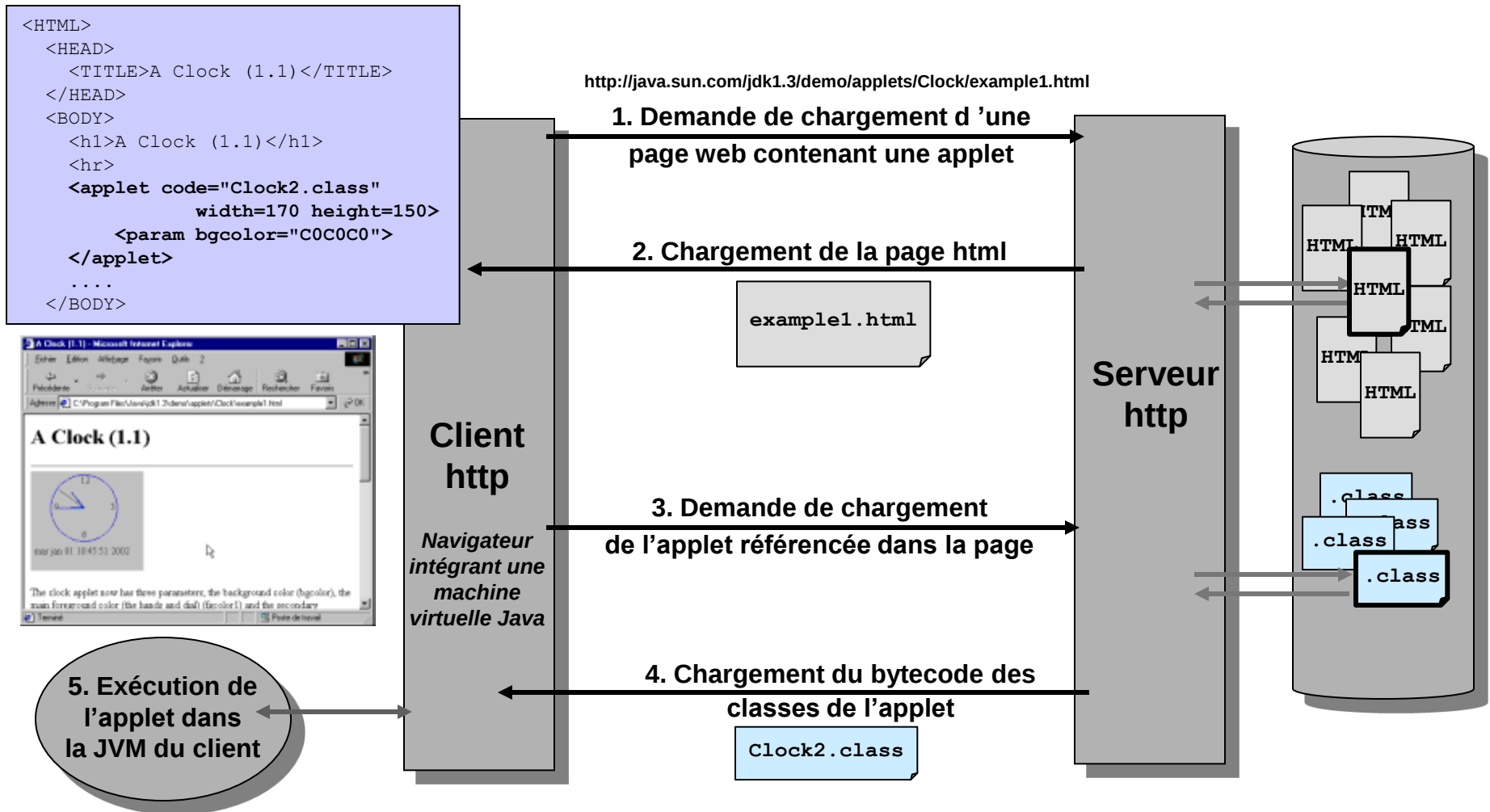
Types de programmes Java

Applet

- Classe principale ne possède pas de méthode `main()`
- Hérite de `java.awt.Applet` ou `javax.swing.JApplet`
- Son bytecode réside sur un serveur http
- Elle est véhiculée vers un client http (navigateur Web) via une page html qui contient son url
- Lorsqu'un navigateur compatible Java (avec sa propre machine virtuelle java (JVM)) reçoit cette page HTML, il télécharge le code de la classe et l'exécute sur le poste client
 - l'applet doit posséder un certain nombre de méthodes pour permettre cette exécution
 - `init()`, `start()`, `stop()`, `paint()`, `destroy()`

Types de programmes Java

Applet : Principe de fonctionnement

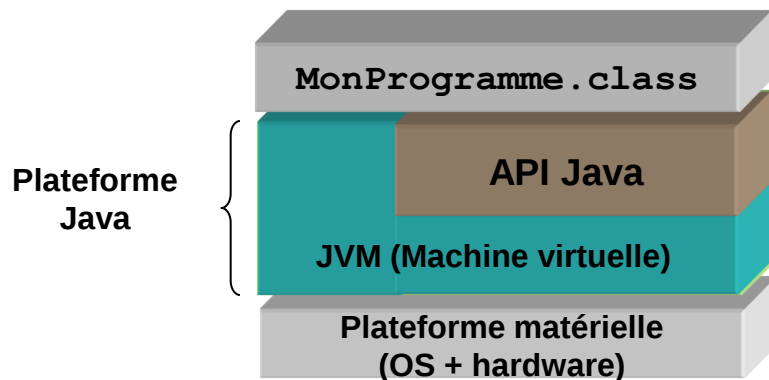


La plateforme Java

- **Plateforme**

- Environnement matériel et/ou logiciel dans lequel un programme s'exécute.
 - La plus part des plateformes sont la combinaison d'un OS et du matériel sous-jacent (*MS Windows + Intel, Linux + Intel, Solaris + Sparc, Mac Os X + Power PC*)

- **La plateforme Java est entièrement logicielle et s'exécute au dessus des plateformes matérielles**

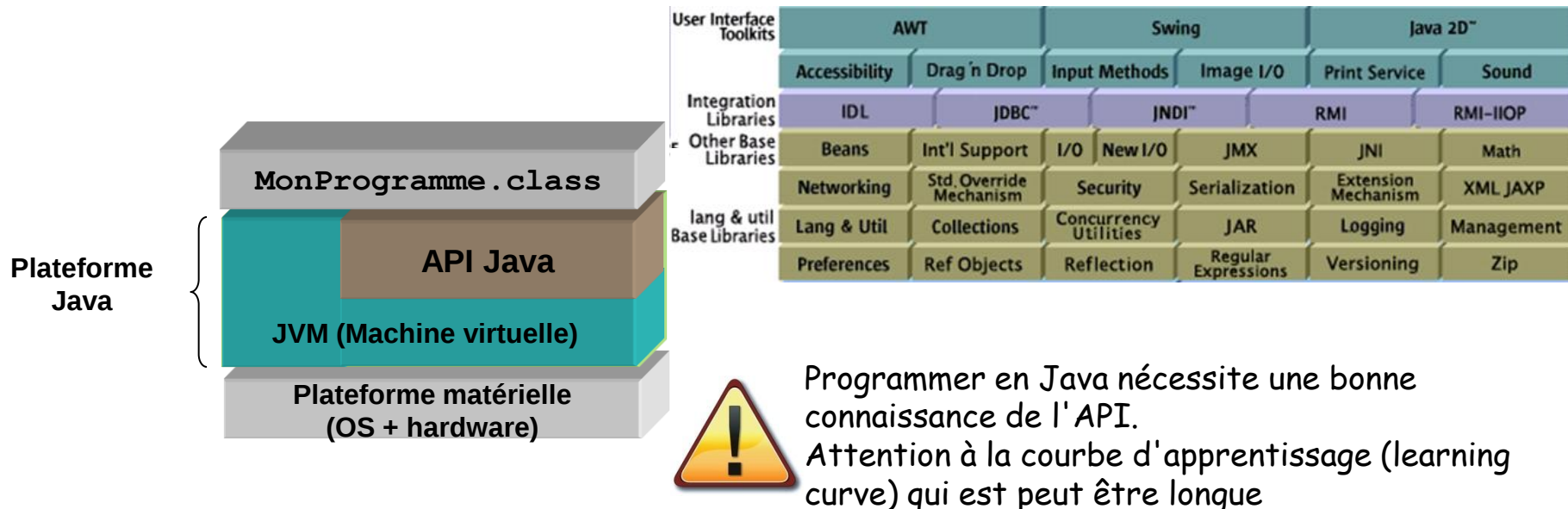


API (Application Programming Interface) Java :
Bibliothèques Java standards sur lesquelles le programmeur peut s'appuyer pour écrire son code

La plateforme Java

API Java

- **API Java**
 - (très) vaste collection de composants logiciels (classes et interfaces)
 - organisée en bibliothèques (**packages**)
 - offre de nombreux services de manière standard (indépendamment de la plateforme matérielle)



La plateforme Java

Les différentes éditions de Java

- 3 éditions de Java



Standard Edition JSE

Fourni les compilateurs, outils, runtimes, et APIs pour écrire, déployer, et exécuter des applets et applications dans la langage de programmation Java



Entreprise Edition JEE

Destinée au développement d'applications « d'entreprise » (*«business applications»*) robustes et interopérables. Simplifier le développement et le déploiement d'applications distribuées et articulées autour du web.



Mobile Edition JME

Environnement d'exécution optimisé pour les dispositifs « légers » :

- Carte à puce (smart cards)
- Téléphones mobiles
- Assistants personnels (PDA)

Versions MS Windows, Linux, Solaris téléchargeables **gratuitement** sur java.sun.com

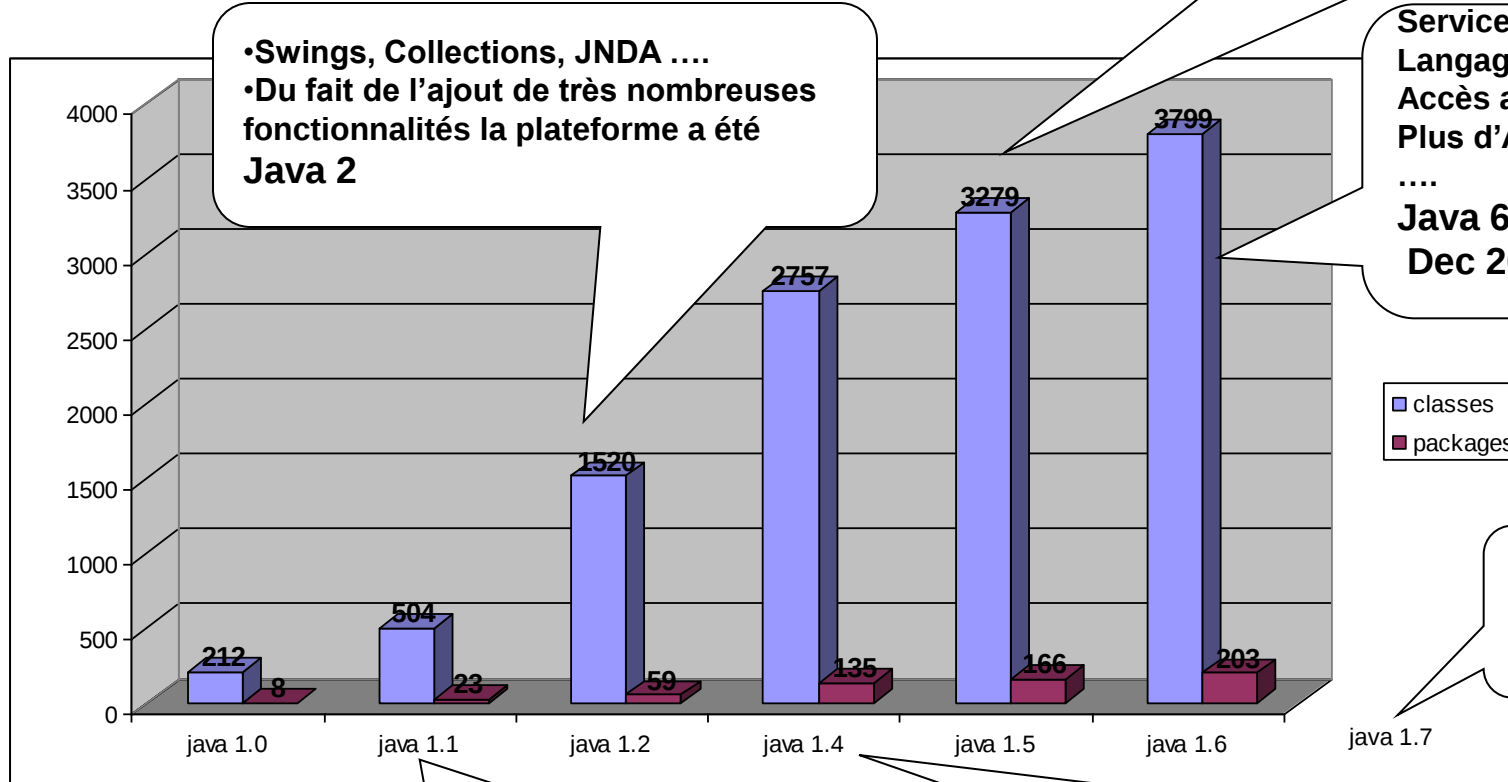
La plateforme Java

Evolution de l'API Standard

•Evolutions du langage : Généricité, Types Enumérés, Autoboxing, ...
•Nouvelles API : Concurrency, Management (gestion et supervision de la JVM)...
Java 5 (Tiger)

•Swings, Collections, JNDA
•Du fait de l'ajout de très nombreuses fonctionnalités la plateforme a été
Java 2

Services Web
Langages de scripts
Accès au compilateur
Plus d'API pour applis desktop
....
Java 6 (Mustang)
Dec 2006



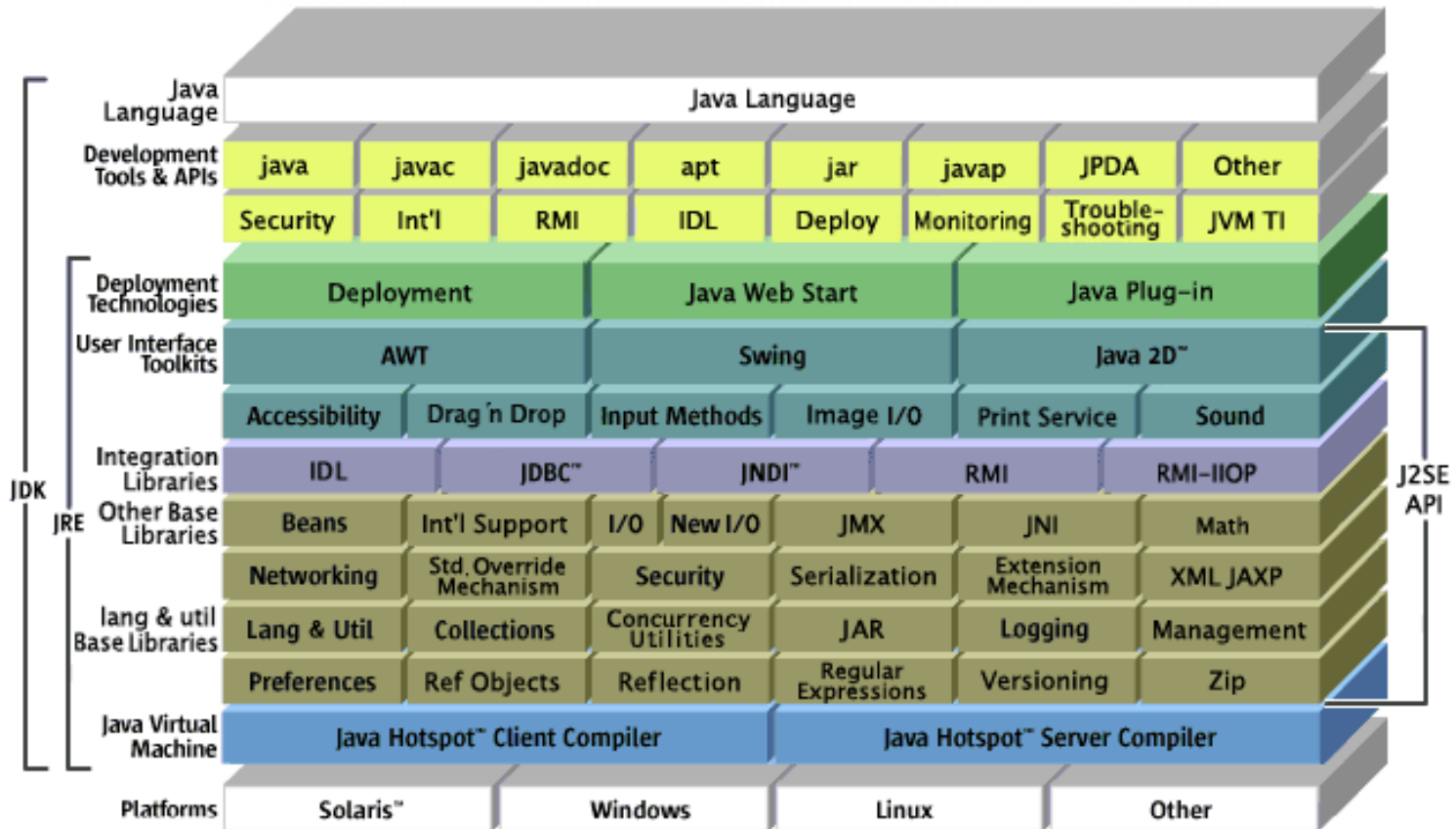
•Modification du langage : classes internes
•Modification du modèle événementiel
•pour awt Java Beans
•Amélioration des performances de la VM

•Amélioration des performances
•Assertions, Nouveau package pour i/o,
•Nouvelles classes pour collections...
•Nombreux nouveaux packages
(XML, log des applications,
préférences utilisateur ...)

Java 7 (Dolphin)
Open Source
2010 ? 2011 ?
2012 ???

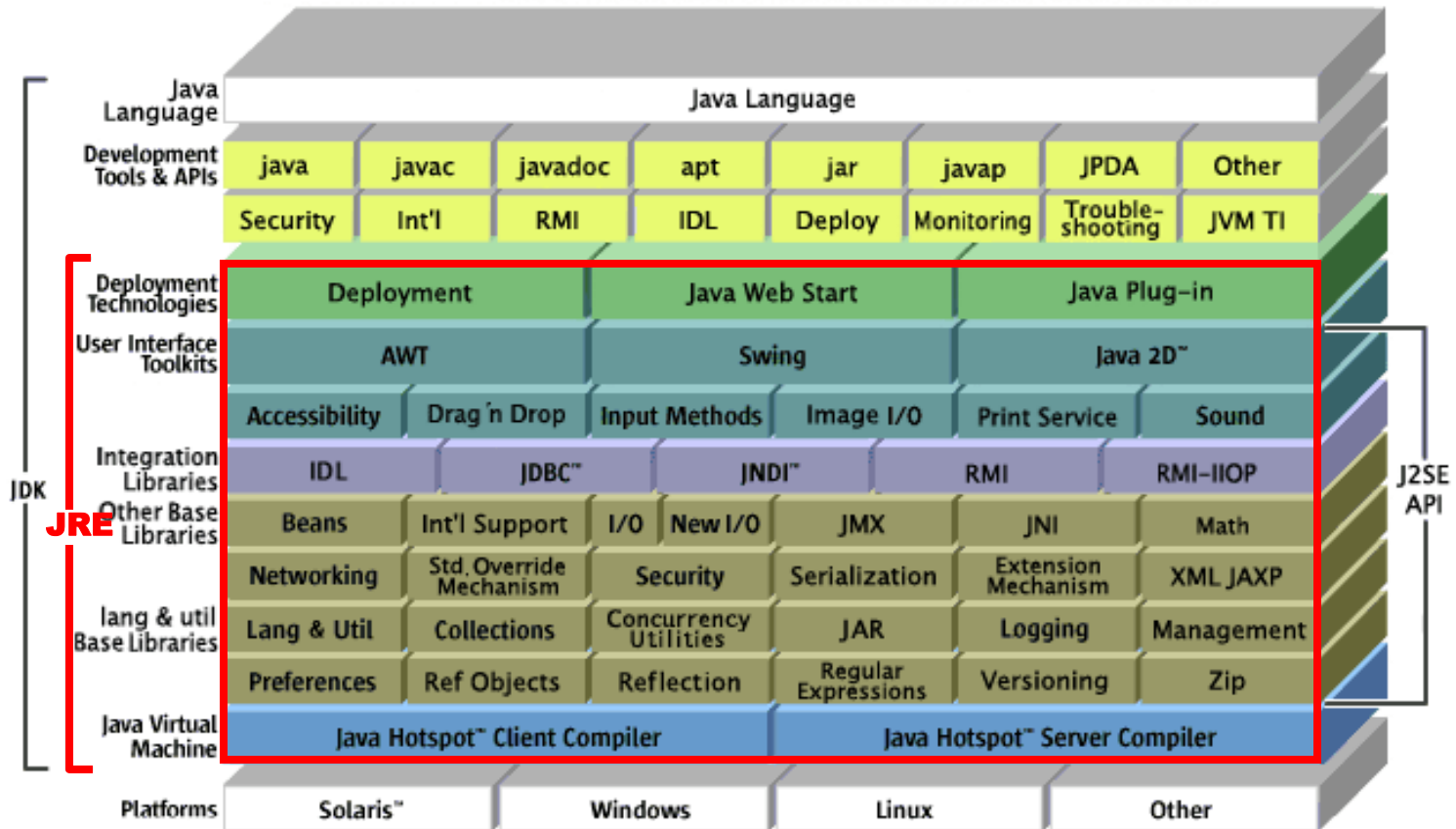
JSE : Java Standard Edition

Organisation générale



JSE : Java Standard Edition

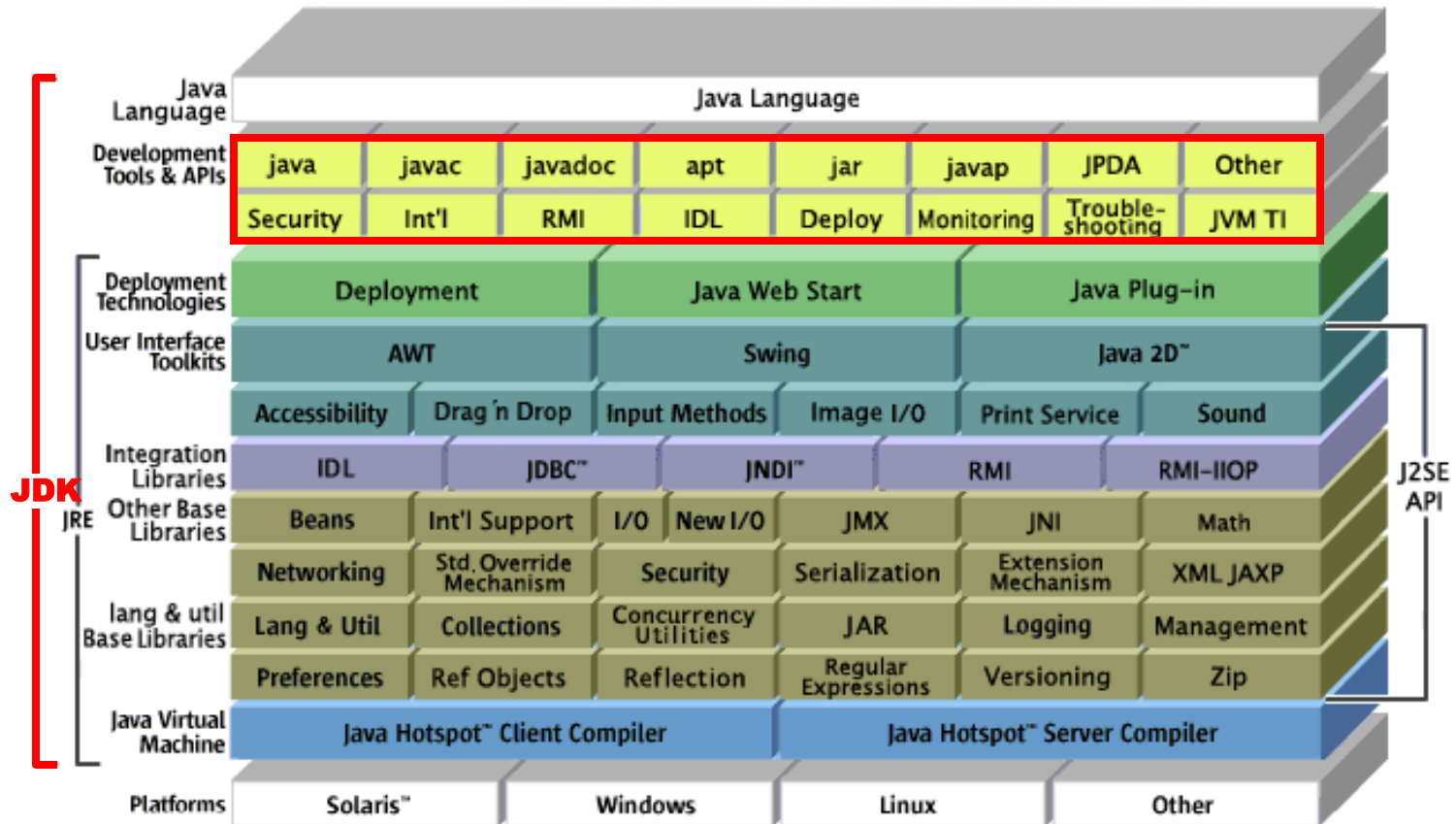
Organisation générale



JRE (Java Runtime Environnement) pour l'exécution de code java compilé

JSE : Java Standard Edition

Organisation générale



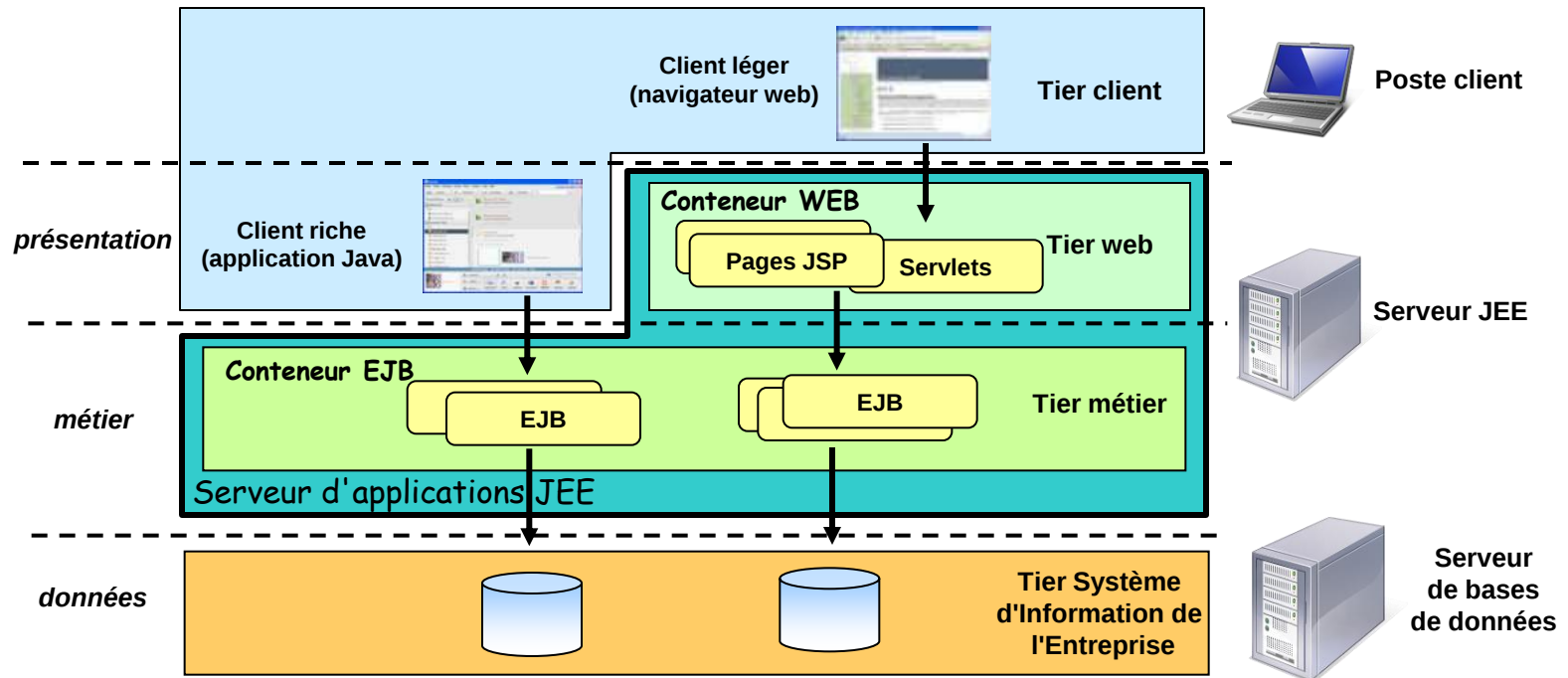
JDK (Java Developer's Kit) outils de base pour le développement d'applications Java

JEE Java Entreprise Edition

- **Plateforme définie par SUN pour le développement d'applications réparties**
 - basée sur un ensemble de technologies Java
 - Servlets, JSP(Java Server Pages), JSF (Java Server Faces) : pages web dynamiques
 - EJB (Entreprise Java Beans) : objets métiers
 - JDBC : API d'accès à des SGBD
 - JNDI (Java Naming and Directory Interface) : API pour nommage des objets
 - JTA (Java Transaction API) ; API pour gestion des transactions
 - JAAS (Java Authentication and Authorization Service)
 - Et de nombreuses autres...
 - S'appuie sur un modèle d'architecture multi-tiers (multi-couches)

Architecture des applications JEE

- Architecture multi-tiers



- Logique de l'application :**
 - Composants web (Servlet, JSP, JFS)
 - Composants métiers (EJB)
- Services standards (cycle de vie des composants, multithreading, transactions, persistance...) pris en charge par les conteneurs Web et EJB du serveur d'application JEE**

JEE situation actuelle

- **Une spécification (Sun) : JEE 5 (2006), JEE 6 (2009)**
- **Différentes implémentations de la plateforme**
 - implémentation de référence
 - (JEE 5 SDK Java Application Platform) fournie gratuitement par Sun
 - Des implémentations commerciales
 - WebLogics (BEA), WebSphere (IBM),
 - Sun Java System Application Server, Oracle Application Server...
 - Ou open-source
 - JBoss, Geronimo (Apache), Glassfish, JOnAS...
- **Processus de certification**
 - TCK (Test Compatibility Kit) (~ 20000 tests)
 - Payant sauf pour plateformes open-source

Environnements de développement intégrés

- **Nombreux IDE (Integrated Development Environment) pour java**
 - Editeur syntaxique, débogueur, compilateur, exécution
- **Commerciaux**



JCreator
Xinox



WebSphere Studio
Site Developer for Java
IBM



JBUILDER
Codegear

ORACLE 10^g
JDEVELOPER

JDeveloper
Oracle

IntelliJIDEA
JetBrains

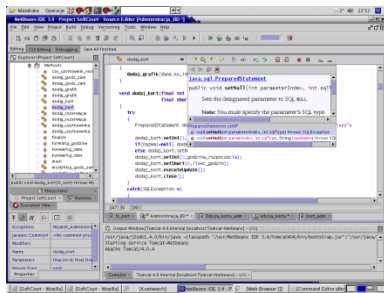
...



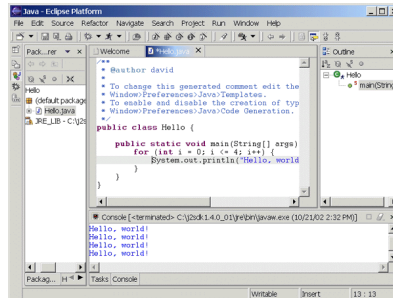
Visual J++
Microsoft

C# .net

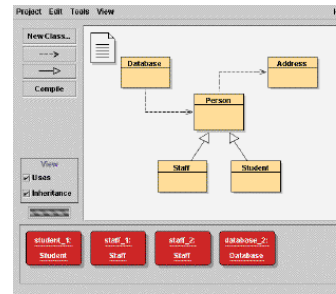
- **Open-source et/ou freeware**



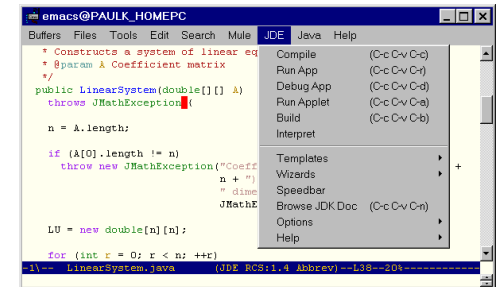
NetBeans
www.netbeans.org



Eclipse
www.eclipse.org



BlueJ
www.bluej.org



Emacs + JDE
<http://sunsite.auc.dk/jde>

Java aujourd'hui

- **Une maturité certaine**
 - Technologie largement reconnue dans le monde industriel
- **Evolutions de JAVA contrôlées par organisation indépendante Java Community Process (www.jcp.org)**
 - Évolutions du langage (généricité dans version 1.5(Tiger))
 - Nouvelles API
- **Passage de Java dans la communauté open source (2007)**
 - Jusqu'à présent JAVA était une marque propriétaire de SUN
- **Bataille avec C# .NET sur la plateforme Windows**
- **Rachat de SUN par Oracle (avril 2009)**

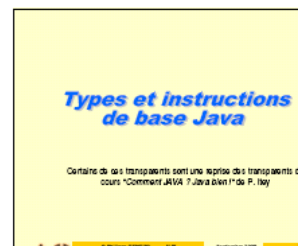
1^{ère} séance de TP

- **utilisation depuis ligne de commande des outils de base pour le développement d'applications Java**
 - Compilation :
 - javac
 - Exécution :
 - java (applications)
 - appletviewer (applets)
 - Documentation
 - javadoc
 - "Archivage"
 - jar
- **Pourquoi ne pas utiliser tout de suite un IDE Java ?**
 - Être capable de travailler avec Java quelque soit l'environnement
 - Comprendre des mécanismes parfois masqués par des environnements intégrés (ex. CLASSPATH)

2^{ème} séance de TP

- **Objectif** : expérimenter les constructions de base du langage Java (types simples, instructions de contrôle, itérations) qui sont très proches syntaxiquement de celles utilisées par le langage C
- **Ecriture de programmes simples**
 - Thème 1 : expressions
 - Thème 2 : instructions conditionnelles
 - Thème 3 : itérations
 - Thème 4 : instanciation d'objets - envois de messages

<http://www.inrialpes.fr/helix/people/genoud/ENSJAVA/cours/>



<http://www.inrialpes.fr/helix/people/genoud/ENSJAVA/cours/Java/typessimples.html>

TYPES ET INSTRUCTIONS ELEMENTAIRES

Types simples

[entiers](#)
[réels](#)
[caractère](#)
[booléen](#)

Enoncés élémentaires

[déclaration de variable](#)
[affectation](#)

Flot de contrôle

[bloc d'instructions](#)
[if ... else](#)
[while](#)
[do while](#)
[for](#)

Affichage

[System.out.println](#)
[System.out.printf](#)

Références

« *Java - tête la première* »

Kathy Sierra, Bert Bates, 2ème édition, O'Reilly, septembre 2005



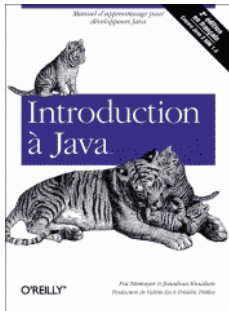
« *Cahiers du Programmeur - Java - 1.4 et 5.0* »

Emmanuel Puybaret, Eyrolles, mars 2006



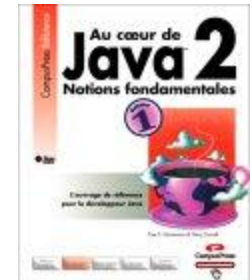
« *Introduction à Java* », 2e édition

Pat Niemeyer et Jonathan Knudsen, O'Reilly, déc. 2002



« *Au coeur de Java 2 - Volume 1 : Notions fondamentales* »

Cay S. Horstmann, Gary Cornell, CampusPress, nov. 2003



« *Thinking in Java* »,

Bruce Eckel - Prentice-Hall (www.BruceEckel.com , www.penserenjava.free)

« *JAVA in a nutshell, 5th Edition* »,

David Flanagan - O'Reilly 2005

...

Références (suite)

URLs

<http://www.oracle.com/technetwork/java/index.html>

- ~~<http://java.sun.com>~~ - Site officiel Java de ~~SUN~~ ORACLE
 - JDK, Tutoriels, Documentations, spécifications, ...
- <http://www.javaworld.com>
Magazine électronique
- <http://www.jguru.com>, <http://www.jdance.com> –
Sites dédiés à la technologie java
 - applets, applications, notes techniques, forums de discussions
- ...
- www.developpez.com
des tutoriels, des FAQ, des ressources....
- <http://penserjava.free.fr>
une traduction du livre de BruceEckel « Thinking in Java »
- ...