

# Java Introduction



Facile



Normal



Difficile



Professionnel



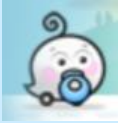
Expert

[https://wiki.waze.com/wiki/Your\\_Rank\\_and\\_Points](https://wiki.waze.com/wiki/Your_Rank_and_Points)

- 1 - Généralités**
- 2 - Les outils et techniques de base**
- 3 - Les bases du langage Java**
- 4 - Bibliographies**



- Les caractéristiques
- Historique
- Indépendance de la plate-forme
- API java
- Package de base
- Catégories des technologies Java
- De java 8 à java 12



Java est un langage de programmation à usage général, évolué et orienté objet dont la syntaxe est proche du C/C++.

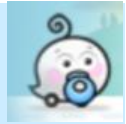
Java est notamment largement utilisé pour le développement d'applications d'entreprises et mobiles.

Quelques chiffres et faits à propos de Java en 2011 :

- 97% des machines d'entreprises ont une JVM installée
- Java est téléchargé plus d'un milliards de fois chaque année
- Il y a plus de 9 millions de développeurs Java dans le monde
- Java est un des langages les plus utilisés dans le monde
- Tous les lecteurs de Blu-Ray utilisent Java
- Plus de 3 milliards d'appareils mobiles peuvent mettre en œuvre Java
- Plus de 1,4 milliards de cartes à puce utilisant Java sont produites chaque année

<http://jmdoudoux.developpez.com/cours/developpons/java/chap-presentation.php>

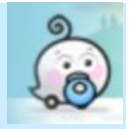
# 1 - Généralités -> Les caractéristiques



<b>Java est interprété</b>	le source est compilé en pseudo code ou bytecode puis exécuté par un interpréteur Java : la Java Virtual Machine (JVM).
<b>Java est portable : il est indépendant de toute plate-forme</b>	il n'y a pas de compilation spécifique pour chaque plate forme. Il est possible d'exécuter des programmes Java sur tous les environnements qui possèdent une Java Virtual Machine.
<b>Java est orienté objet.</b>	Chaque fichier source contient la définition d'une ou plusieurs classes. Java n'est pas complètement objet car il définit des types primitifs (entier, caractère, flottant, booléen ...).
<b>Java est simple</b>	le choix de ses auteurs a été d'abandonner des éléments mal compris ou mal exploités des autres langages tels que la notion de pointeurs, l'héritage multiple et la surcharge des opérateurs ...
<b>Java est fortement typé</b>	toutes les variables sont typées et il n'existe pas de conversion automatique qui risquerait une perte de données.
<b>Java assure la gestion de la mémoire</b>	l'allocation de la mémoire pour un objet est automatique à sa création et Java récupère automatiquement la mémoire inutilisée grâce au garbage collector
<b>Java est économe</b>	le pseudo code a une taille relativement petite car les bibliothèques de classes requises ne sont liées qu'à l'exécution.
<b>Java est multitâche</b>	il permet l'utilisation de threads qui sont des unités d'exécutions isolées. La JVM, elle même, utilise plusieurs threads.

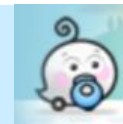
<http://jmdoudoux.developpez.com/cours/developpons/java/chap-presentation.php>

# 1 - Généralités -> Historique



Année	Evénements
<b>1995</b>	mai : premier lancement commercial du JDK 1.0
<b>1996</b>	janvier : JDK 1.0.1 septembre : lancement du JDC
<b>1997</b>	Java Card 2.0 février : JDK 1.1
<b>1998</b>	décembre : lancement de J2SE 1.2 et du JCP Personal Java 1.0
<b>1999</b>	décembre : lancement J2EE 1.2
<b>2000</b>	mai : J2SE 1.3
<b>2001</b>	J2EE 1.3
<b>2002</b>	février : J2SE 1.4
<b>2003</b>	J2EE 1.4
<b>2004</b>	septembre : J2SE 5.0
<b>2005</b>	Lancement du programme Java Champion
<b>2006</b>	mai : Java EE 5 décembre : Java SE 6.0
<b>2007</b>	Duke, la mascotte de Java est sous la licence Free BSD
<b>2008</b>	décembre : Java FX 1.0
<b>2009</b>	février : JavaFX 1.1 juin : JavaFX 1.2 décembre : Java EE 6
<b>2010</b>	janvier : rachat de Sun par Oracle avril : JavaFX 1.3
<b>2011</b>	juillet : Java SE 7 octobre : JavaFX 2.0
<b>2012</b>	août : JavaFX 2.2
<b>2013</b>	juin : Java EE 7

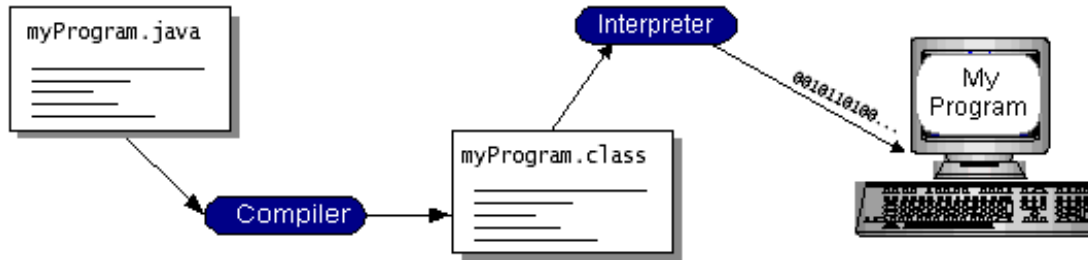
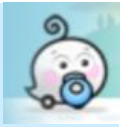
# 1 - Généralités -> Historique



Année	Evénements
03/2014	Java 8
09/2017	Java 9
03/2018	Java 10
09/2018	java 11
03/2019	Java 12
09/2019	Java13
03/2020	Java 14
09/2020	Java 15
03/2021	Java 16
09/2021	Java 17

De nouvelles versions de Java tous **les six mois** depuis la sortie de Java 9 en septembre 2017  
=> Java 9 date de moins de 3 ans, la dernière version est maintenant Java 17. ( java 18 prévue le 03/2021)

# 1 - Généralités -> Indépendance de la plate-forme

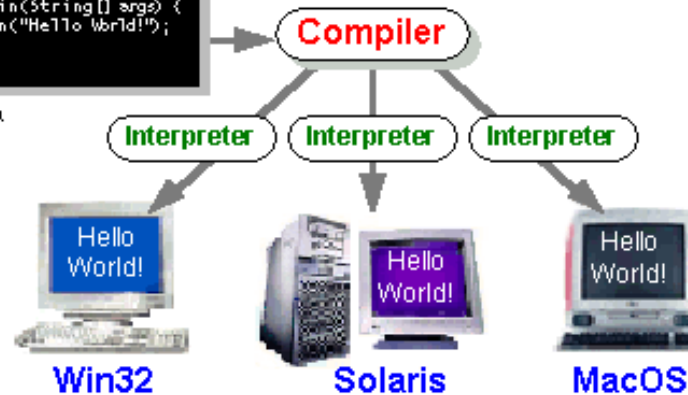


## Java Program

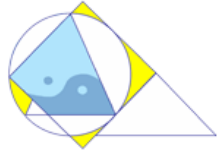
```

class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
  
```

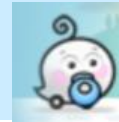
HelloWorldApp.java



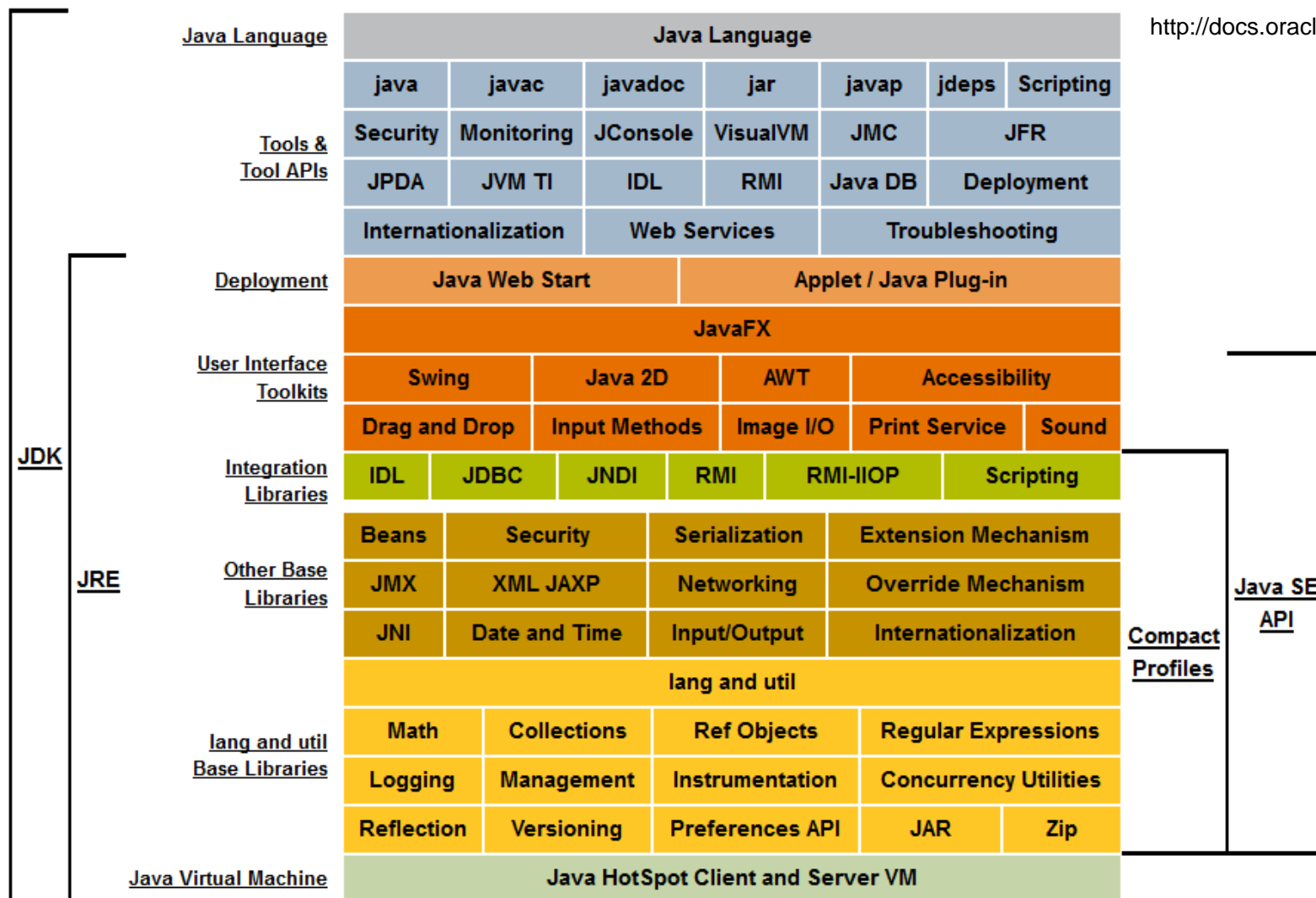




# 1 - Généralités -> API java



<http://docs.oracle.com/javase/8/docs/>



# 1 - Généralités -> Package de base

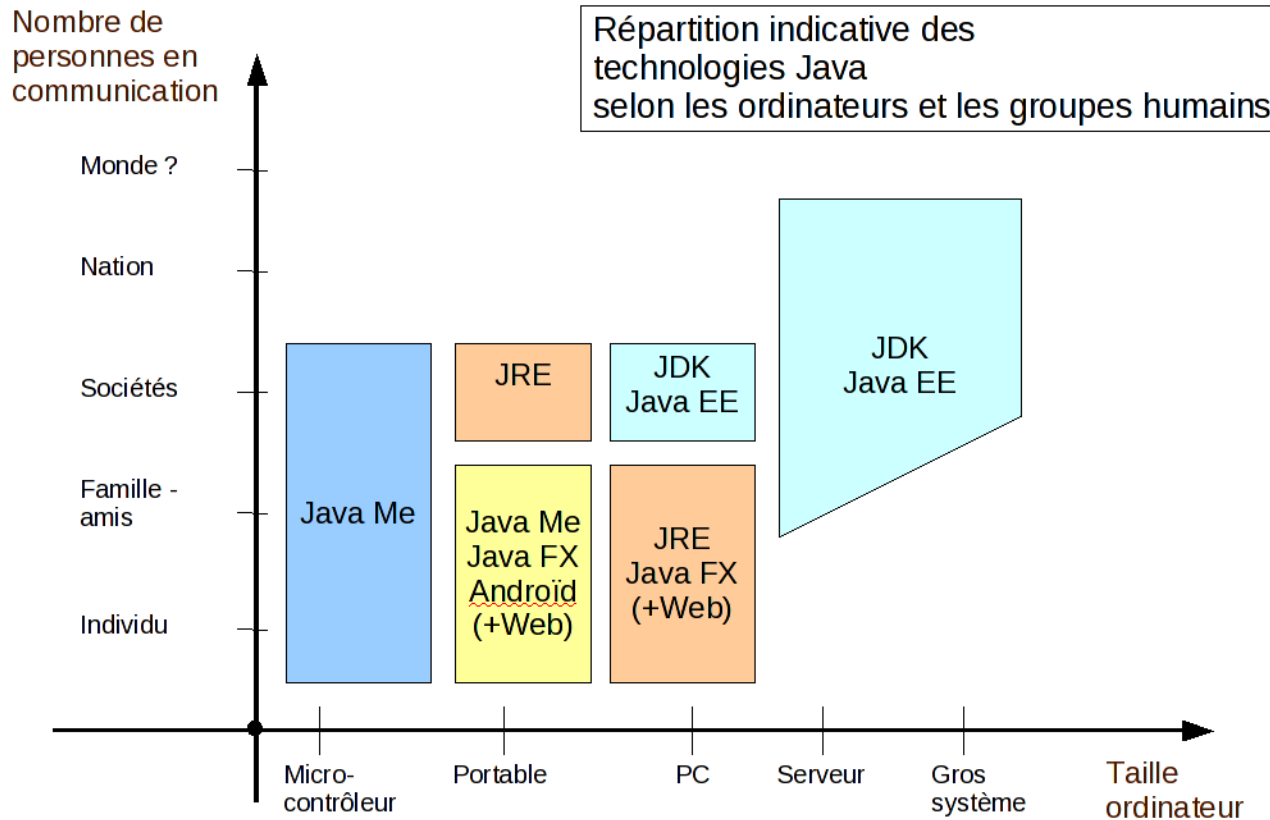


	Java 1.0	Java 1.1	Java 1.2	J2SE 1.3	J2SE 1.4	J2SE 5.0	Java SE 6	Java SE 7	Java SE 8
<b>Nombre de packages</b>	8	23	59	76	135	166	202	209	217
<b>Nombre de classes</b>	201	503	1520	1840	2990	3280	3780	4024	4240



- **Java Platform, Standard Edition (Java SE)**
  - **Java Platform, Enterprise Edition (Java EE)**
  - **JAVA PLATFORM, MICRO EDITION (JAVA ME)**
  - **JavaFX** contient des outils très divers, notamment pour les médias audios et vidéos, le graphisme 2D et 3D, la programmation Web, la programmation multi-fils etc.
  - **Le JRE** qui se compose d'une machine virtuelle, de bibliothèques logicielles utilisées par les programmes Java et d'un plugin pour permettre l'exécution de ces programmes depuis les navigateurs web.
- 
- ORACLE JAVA EMBEDDED
  - Java DB is Oracle's supported distribution of the Apache Derby open source database.
  - JAVA CARD TECHNOLOGY
  - JAVA TV

# 1 - Généralités -> Catégories des technologies Java



# 1 - Généralités -> De java 8 à java 14

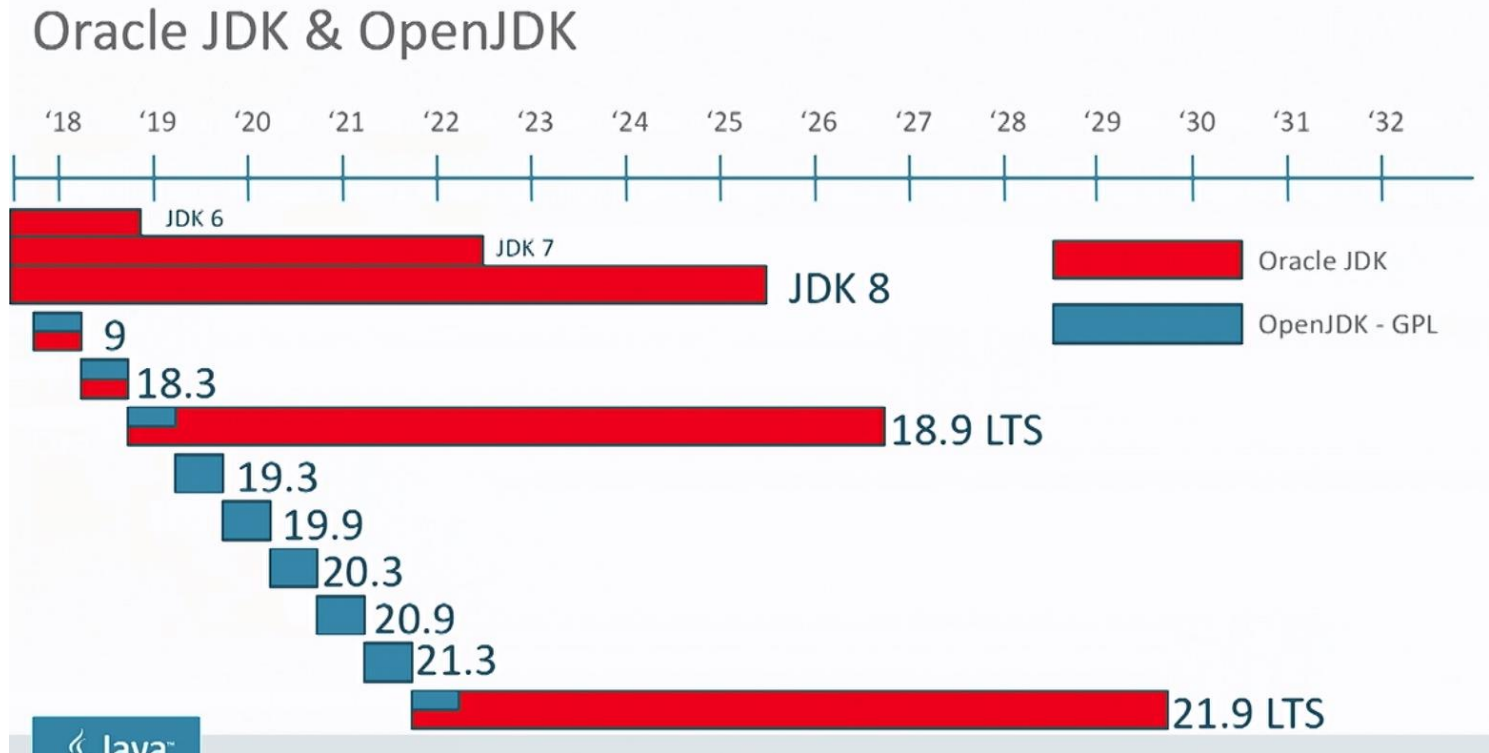
Version	Nouveauté
Java 8 03/2014	Ajout des <i>lambdas</i> , entraînant une refonte d'une partie de l'API, notamment les collections et la notion de <i>stream</i> . Les autres ajouts notables incluent les <b>Optional</b> , les implémentations par défaut au sein d'une interface, une refonte de l'API date, etc.
Java 9 09/2017	Le projet <b>Jigsaw</b> permettant de modulariser les modules chargés au sein du JDK ; Le projet <b>Kulla</b> visant la création d'un shell pour Java sur le format read-eval-print loop Le projet <b>Valhalla</b> visant une amélioration des types Java ; Un support natif du format JSON et de HTTP/253.
Java 10 03/2018	Inférence des types des variables locales ( <a href="https://www.baeldung.com/java-10-local-variable-type-inference">https://www.baeldung.com/java-10-local-variable-type-inference</a> ) Partage de binaire pour permettre un lancement plus rapide Activation de Graal un compilateur JIT en Java

# 1 - Généralités -> De java 8 à java 14

Version	Nouveauté
<b>Java 11</b> <b>09/2018</b>	Amélioration sur les paramètres des lambda Un client HTTP plus évolué Suppression des modules CORBA et EE par défaut
<b>Java 12</b> <b>03/2019</b>	Shenandoah: Un ramasse miette avec de courtes pauses (Expérimentale) Suite d'outils de Microbenchmark pour le code source du JDK Expressions Switch API Constants (permettre d'ajouter des informations dans les méta données dans les fichiers .class, utile pour les langages sur la JVM) Un seul portage pour l'architecture ARM 64bits Default CDS Archives (chargement des informations des classes de la JVM plus rapide) Améliorations du ramasse miette G1
<b>Java 13</b> <b>09/2019</b>	Deux grosses nouveautés pour les développeurs sont apparues sur Java 13, les blocs de texte ainsi que l'apparition du mot clé « yield » sur les expressions switch
<b>Java 14</b> <b>03/2020</b>	Amélioration des Text Blocks et Des NullPointerException

<https://zenika.developpez.com/tutoriels/java/comprendre-nouveautes-java12-a-java14/>

OpenJDK est l'implémentation libre (sous licence GPL) de la plateforme Java SE d'Oracle. Elle se compose de plusieurs contributeurs dont Oracle, Red Hat, Azul Systems, SAP SE, IBM, Apple ....



<https://webdevdesigner.com/q/differences-between-oracle-jdk-and-openjdk-48296/>

<http://openjdk.java.net/install/>

<https://tekcollab.imdeo.com/java-devient-payant/>

[https://www.youtube.com/watch?v=gV5T4AMFS\\_A](https://www.youtube.com/watch?v=gV5T4AMFS_A)

# 1 - Généralités -> Quelle version de java utilisez ?

## Quelle version de Java utilisez-vous ?

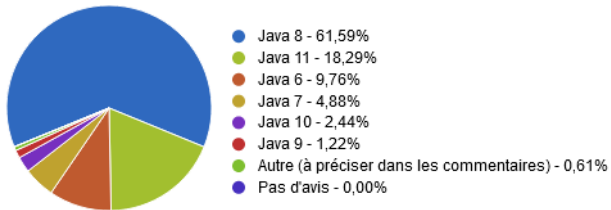
Qu'est-ce qui vous empêche de migrer vers une version plus récente ?

Le 13 mars 2019 à 14:44, par [Michael Guilloux](#) | 22 commentaires



1.2K PARTAGES

Quelle version de Java utilisez-vous ?



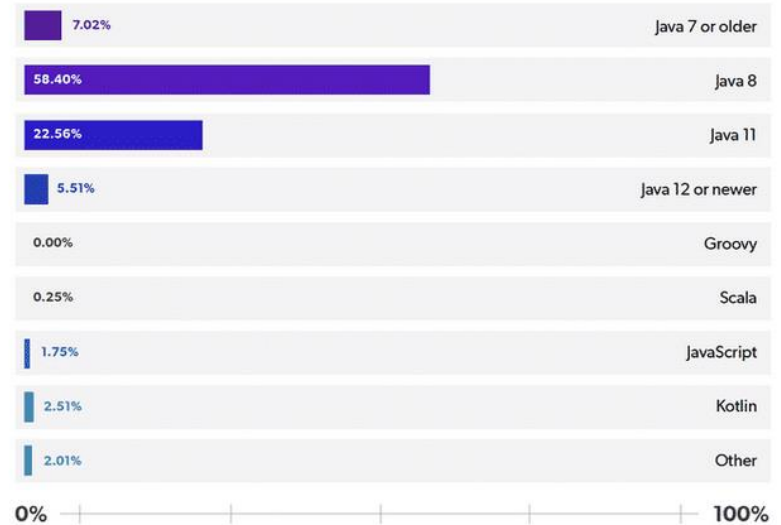
164 votants

Voter

## Java : les développeurs privilégient Java 8 et Tomcat

Ariane Beky, 25 février 2020, 19:08

What Java programming language are you using in your main application?



<https://www.developpez.com/actu/250988/Quelle-version-de-Java-utilisez-vous-Qu'est-ce-qui-vous-empêche-de-migrer-vers-une-version-plus-récente/>

<https://www.silicon.fr/java-developpeurs-java8-tomcat-334720.html>



# 1 - Généralités -> Quelle version de java utilisez ?

**Silicon** CLOUD BLOCKCHAIN CYBERSÉCURITÉ MOBILITÉ / RESEAUX WORKPLACE L

charge (mises à jour et maintenance) devait arriver à son terme fin 2025.

Pour quelles raisons Oracle a prolongé le support de cette version de Java ?

**Java 8, sorti en mars 2014, toujours apprécié**

Malgré ses six années de service et de nombreux successeurs ([Java 14](#) vient de sortir), Java 8 est la version la plus utilisée de la technologie, à la fois langage de programmation orienté objet et plateforme informatique (Java Development Kit, JDK).

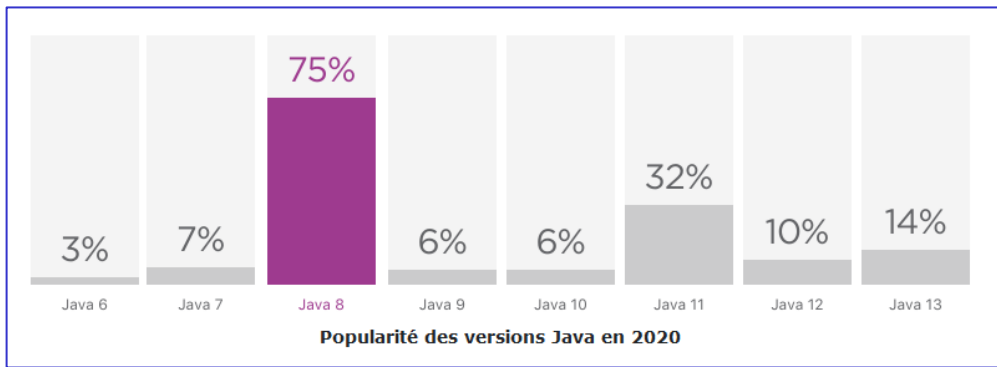
Selon un rapport de Perforce Software, 58% des [développeurs Java](#) interrogés l'automne dernier s'appuyaient sur Java 8, tandis que 22,5% s'étaient tournés vers Java 11. Une minorité privilégiant d'autres versions, antérieures ou plus récentes. Oracle, de son côté, estime que 30 à 40% utilisent Java 11 ou une version plus récente en production, actuellement. Java 8 restant encore la version la plus populaire, et de loin.

Java 8 est une version couplée à un support à long terme (LTS), comme la version 11 (jusqu'en 2026, à ce jour). Les éditions plus récentes du programme ne bénéficient pas de cette prise en charge étendue, mais d'un support initial de six mois.

<https://www.silicon.fr/oracle-rallonge-support-java-8-336554.html>

## Oracle rallonge le support du très populaire Java 8

Ariane Beky, 19 mars 2020, 17:55 | Mis à jour le 3 janvier 2022, 16:24



<https://jetbrains.developpez.com/actu/309040/Les-chiffres-cles-de-la-communaute-Java-types-de-logiciels-developpes-secteurs-d-activite-versions-EDI-framework-JetBrains-dresse-le-portrait-du-langage-en-2020/>

<https://www.silicon.fr/oracle-rallonge-support-java-8-336554.html>



- plate-forme de développement
- la gestion des dépendances

<https://java.developpez.com/telecharger/index/categorie/336/RAD-et-EDI-Java>

## 2 - Les outils et techniques de base

### -> plate-forme de développement



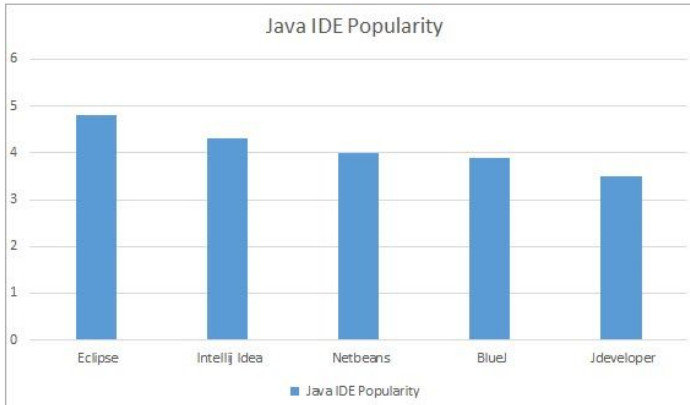
- **IntelliJ IDEA** est un IDE Java commercial développé par Jettersais.  
<https://www.jetbrains.com/idea/>
  - **Android Studio** est un environnement de développement pour développer des applications Android. Il est basé sur IntelliJ IDEA.
- **Netbeans** est un environnement de développement open source écrit en Java. Le produit est composé d'une partie centrale à laquelle il est possible d'ajouter des modules. [netbeans.org/](https://netbeans.org/)
- **Eclipse** est un projet open source à l'origine développé par IBM pour ses futurs outils de développement et offert à la communauté.  
<http://www.eclipse.org>
- **Oracle JDeveloper** est un EDI complet et gratuit permettant de modéliser et développer des applications Java pour les plateformes J2SE, J2EE ou J2ME.

<https://waytolearnx.com/2020/04/top-10-des-meilleurs-ide-pour-les-developpeurs-java.html>

## 2 - Les outils et techniques de base -> plate-forme de développement

### Graphique des 5 meilleurs logiciels Java IDE

Le graphique ci-dessous montre la popularité des 5 meilleurs IDE Java.



IDE Java	Note de l'utilisateur	Satisfaction des utilisateurs	Échelle de courbe d'apprentissage	Mise en évidence de la syntaxe	Performance
Éclipse	4,8 / 5	92%	Facile	Oui	Bien
Idée IntelliJ	4,3 / 5	89%	Moyen	Oui	Moyen
NetBeans	4.1 / 5	85%	Moyen	Non	Moyen
JDeveloper	4/5	80%	Facile	Oui	Moyen
Studio Android	4,3 / 5	90%	Tremper	Non	Bien
BLUEJ	4.1	82%	Moyen	Oui	Moyen

<https://fr.myservername.com/top-10-best-java-ides-online-java-compilers>

# Démarrer avec intellij-idea

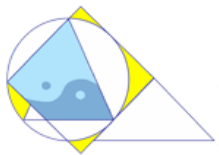
- <https://riptutorial.com/fr/intellij-idea>
- <https://www.jetbrains.com/help/idea/creating-and-running-your-first-java-application.html>

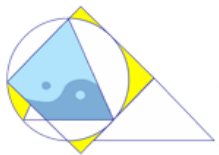
# Documentation

- <https://www.jetbrains.com/fr-fr/idea/documentation/>



- **Junit** et **FestAssert** permettent d'écrire les tests unitaires.
- **Mockito** et **EasyMock** outils pour tester les classes en isolation
- Le trio **PMD/Findbugs/Checkstyle** qui permet de vérifier la qualité du code et publier les rapports obtenus
- **Apache CXF** est un framework open-source en langage Java, facilitant le développement de services web
- **Spring est un framework libre** qui permet de construire et de définir l'infrastructure d'une application java, dont il facilite le développement et les tests







## 3 - Les bases du langage Java



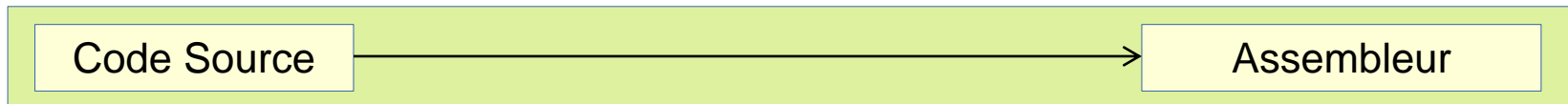
- La compilation à l'exécution
- Mots-clés du java
- Types prédéfinis
- Structure Lexicale
- Les commentaires
- Définition des types de base
- Les opérateurs
- Les structures de contrôle et boucle
- Les chaînes de caractères
- Les tableaux
- Les conversions de types
- La manipulation des chaînes de caractères
- Références
- Passage par valeur ou par référence

### 3 - Les bases du langage Java

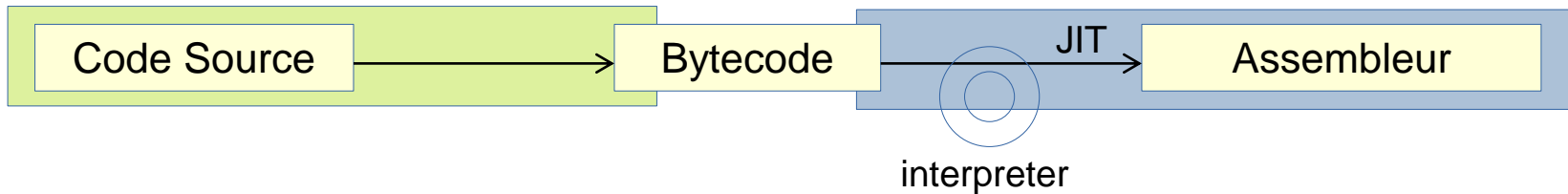
#### -> La compilation à l'exécution : Le bytecode portable

<http://www-igm.univ-mlv.fr/~forax/>

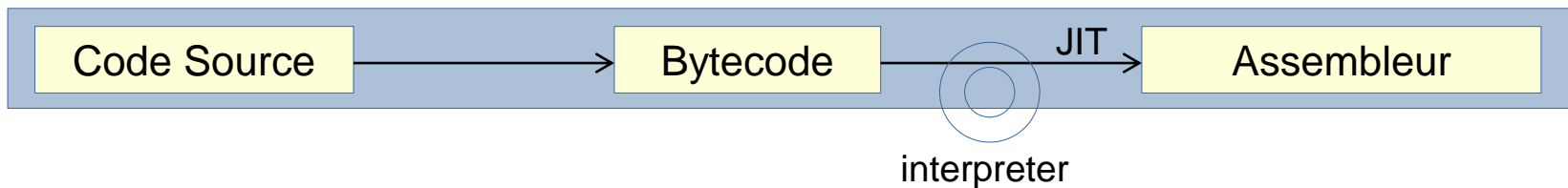
#### Modèle du C



#### Modèle de Java



#### Modèle de JavaScript



A la compilation

A l'exécution

## 3 - Les bases du langage Java

### -> La compilation à l'exécution



HelloWorld.java

**javac** HelloWorld.java

HelloWorld.class

**java** HelloWorld

```
Start Page x HelloWorld.java x
Source History
1  /**
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6   package helloworld;
7
8   /**
9    *
10   * @author Palermo David
11   */
12   public class HelloWorld {
13
14       /**
15        * @param args the command line arguments
16        */
17       public static void main(String[] args) {
18           System.out.println("Hello World ( Bonjour le monde )");
19       }
20
21   }
22
```

```
Output - HelloWorld (run) x
run:
Hello World ( Bonjour le monde )
BUILD SUCCESSFUL (total time: 0 seconds)
```



## Compiler :

**`javac -d build/classes/helloworld src/helloworld/HelloWorld.java`**  
*crée le fichier HelloWorld.class dans build/classes/helloworld*

## Exécuter :

**`java -classpath build\classes helloworld.HelloWorld`**

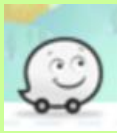
*On indique où se trouve les fichiers binaires (avec -classpath) ainsi que le nom de la classe qui contient le main.*

<https://koor.fr/Java/Tutorial/Compilation.wp>

[https://gayerie.dev/epsi-b3-java/langage\\_java/compilation.html](https://gayerie.dev/epsi-b3-java/langage_java/compilation.html)

## 3 - Les bases du langage Java

### -> Mots-clés du java



Les mots-clés pour les **objets** :

<b>abstract</b>	<b>class</b>	<b>enum</b>	<b>extends</b>	<b>implements</b>	<b>import</b>	<b>interface</b>	<b>native</b>	<b>package</b>	<b>super</b>	<b>this</b>
-----------------	--------------	-------------	----------------	-------------------	---------------	------------------	---------------	----------------	--------------	-------------

Les mots-clés pour les **types** :

<b>boolean</b>	<b>byte</b>	<b>char</b>	<b>double</b>	<b>float</b>	<b>int</b>	<b>long</b>	<b>short</b>	<b>void</b>
----------------	-------------	-------------	---------------	--------------	------------	-------------	--------------	-------------

Les mots-clés pour les **états** :

<b>const</b>	<b>false</b>	<b>final</b>	<b>new</b>	<b>null</b>	<b>static</b>	<b>strictfp</b>	<b>transient</b>	<b>true</b>	<b>volatile</b>
--------------	--------------	--------------	------------	-------------	---------------	-----------------	------------------	-------------	-----------------

Les mots-clés pour les **modificateurs** :

<b>private</b>	<b>protected</b>	<b>public</b>
----------------	------------------	---------------

Les mots-clés pour les **boucles** :

<b>continue</b>	<b>do</b>	<b>for</b>	<b>goto</b>	<b>while</b>
-----------------	-----------	------------	-------------	--------------

Les mots-clés pour les **branchements** :

<b>assert</b>	<b>break</b>	<b>case</b>	<b>default</b>	<b>else</b>	<b>if</b>	<b>instanceof</b>	<b>return</b>	<b>switch</b>	<b>synchronized</b>
---------------	--------------	-------------	----------------	-------------	-----------	-------------------	---------------	---------------	---------------------

Les mots-clés pour les **exceptions** :

<b>catch</b>	<b>finally</b>	<b>throw</b>	<b>throws</b>	<b>try</b>
--------------	----------------	--------------	---------------	------------

### 3 - Les bases du langage Java

#### -> Types prédéfinis : numériques



Type	Description	Taille	Valeur minimale	Valeur maximale
<b>byte</b>	Tout petit entier signé	1	$-2^7$ ou -128	$2^7-1$ ou 127
<b>short</b>	Petit entier signé	2	$-2^{15}$ ou -32768	$2^{15}-1$ ou 32767
<b>int</b>	Entier signé	4	-2147483648	2147483647
			$-2^{31}$ ou -2147483648	ou $2^{31}-1$
<b>long</b>	Grand entier signé	8	$-2^{63}$ ou -9223372036854770000	$2^{63}-1$ ou 9223372036854770000
<b>float</b>	Nombre à virgule flottante	4	$-3,40282347 \times 10^{38}$	$-1,40239846 \times 10^{-45}$
			$1,40239846 \times 10^{-45}$	$3,40282347 \times 10^{38}$
<b>double</b>	Nombre à virgule flottante double précision	8	$-1,79769313486231570 \times 10^{308}$	$-4,94065645841246544 \times 10^{-324}$
			$4,94065645841246544 \times 10^{-324}$	$1,79769313486231570 \times 10^{308}$

☞ La taille est fixe, indépendante de la plate-forme

### 3 - Les bases du langage Java

#### -> Types prédéfinis : non-numériques



Type	Description
<b>boolean</b>	Les variables de type <b><i>boolean</i></b> sont des variables qui ne peuvent contenir que les valeurs : <i>true</i> ou <i>false</i> . il n'est pas possible en Java de substituer un nombre entier à une expression conditionnelle comme en C ou C++
<b>char</b>	Les variables de type char contiennent des valeurs correspondant à des caractères du standard UNICODE. Les éléments de type char occupent 2 octets de mémoire chacun.
<b>void</b>	type vide



- Format libre : blanc, espace, tabulation ignorés
- Jeu de caractère Unicode : ASCII sur 16 bits
- Commentaires
  - *// commentaire sur une ligne*
  - */\* commentaire sur plusieurs lignes \*/*
  - */\*\* ... \*/ pour javadoc*
- Identificateurs
  - Noms de variables ou de classes
  - Commençant par une lettre, suivi de lettres et chiffres





**Javadoc** est un outil développé permettant de créer une documentation d'API en format HTML depuis les commentaires présents dans un code source en Java ( voir aussi doxygen ).

Attribut et syntaxe	Dans un commentaire de ...	Description
@author <i>auteur</i>	classe	Nom de l'auteur de la classe.
@version <i>version</i>	classe	Version de la classe.
@deprecated <i>description</i>	classe, constructeur, méthode, champ	Marquer l'entité comme obsolète (ancienne version), décrire pourquoi et par quoi la remplacer.
@see <i>référence</i>	classe, constructeur, méthode, champ	Ajouter un lien dans la section "Voir aussi".
@param <i>description de l'id</i>	constructeur et méthode	Décrire un paramètre de méthode.
@return <i>description</i>	méthode	Décrire la valeur retournée par une méthode.
@exception <i>description du type</i>	constructeur et méthode	Décrire les raisons de lancement d'une exception du type spécifié (clause throws).

## 3 - Les bases du langage Java

### -> Les commentaires : javadoc



```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package helloworld;

/**
 * Premier classe java
 * @author Palermo David
 */
public class HelloWorld {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        System.out.println("Hello World ( Bonjour le monde )");
    }

    /**
     * Obtenir la somme de deux entiers.
     * @param a Le premier nombre entier.
     * @param b Le deuxième nombre entier.
     * @return La valeur de la somme des deux entiers spécifiés.
     */
    public int somme(int a, int b) {
        return a + b;
    }
}
```



- Comme champ dans une classe :
  - initialisé par défaut : false ou 0
- Comme variable locale :
  - pas de valeur par défaut  $\Rightarrow$  il faut les initialiser ou les affecter.
- Syntaxe : `<type> <identificateur>`  
`int i = 0;`  
`boolean estVrai = false;`  
`double rayon ;`  
`rayon = 1.5;`



- Listes des operateurs
- Priorité des opérateurs
- Les opérateurs : arithmétiques
- Les opérateurs : binaires
- Les opérateurs : conditionnels
- Les opérateurs : simplifiés
- Les opérateurs : logiques

## 3 - Les bases du langage Java

### -> Les opérateurs : Listes des opérateurs



<u>affectation</u>	<u>incrémentation décrémentation</u>	<u>arithmétique</u>	<u>logique</u>	<u>comparaison</u>	<u>accès aux membre</u>	<u>autre</u>
a = b	++a	+a	!a	a == b	a[b]	a(...)
	--a	-a	a && b	a != b	a.b	a, b
a += b	a++	a + b	a    b	a < b		(type) a
a -= b	a--	a - b		a > b		? :
a *= b		a * b		a <= b		
a /= b		a / b		a >= b		
a %= b		a % b				
a &= b		~a				
a  = b		a & b				
a ^= b		a   b				
a <<= b		a ^ b				
a >>= b		a << b				
>>>=		a >> b				
		a>>> b				

## 3 - Les bases du langage Java

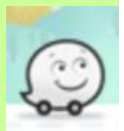
### -> Les opérateurs : Listes des opérateurs



Symbole	Note	Priorité	Associativité
++a --a	Préincrémentation, prédécrémentation	16	Droite à gauche
a++ a--	Postincrémentation, postdécrémentation	15	Gauche à droite
~	Inversion des bits d'un entier	14	Droite à gauche
!	Non logique pour un booléen		Droite à gauche
- +	Moins et plus unaire		Droite à gauche
(type)	Conversion de type (cast)	13	Droite à gauche
* / %	Opérations multiplicatives	12	Gauche à droite
- +	Opérations additives	11	Gauche à droite
<< >> >>>	Décalage de bits, à gauche et à droite	10	Gauche à droite
instanceof < <= > >=	Opérateurs relationnels	9	Gauche à droite
== !=	Opérateurs d'égalité	8	Gauche à droite
&	Et logique bit à bit	7	Gauche à droite

## 3 - Les bases du langage Java

### -> Les opérateurs : Listes des opérateurs



Symbole	Note	Priorité	Associativité
^	Ou exclusif logique bit à bit	6	Gauche à droite
	Ou inclusif logique bit à bit	5	Gauche à droite
&&	Et conditionnel	4	Gauche à droite
	Ou conditionnel	3	Gauche à droite
? :	Opérateur conditionnel	2	Droite à gauche
= *= /= %= += -= <<= >>= >>>= &= ^=	Opérateurs d'affectation	1	Droite à gauche



- Sur les réels : **+** **-** **/** **\***
- Sur les entiers : **+** **-** **/** (quotient de la division), **%** (reste de la division)

l'affectation se fait grâce au signe '='

Hiérarchie habituelle (\* et / prioritaires par rapport aux + et -)





- $\&$ , bitand ( et)

Table de vérité AND

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

- $|$ , bitor (ou)

Table de vérité OR

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

- $\wedge$ , xor (ou exclusif)

Table de vérité XOR

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0



- $\&=$ , `and_eq`, //  $a \&= b \Leftrightarrow a = a \& b$
- $|=$ , `or_eq`, //  $a | b \Leftrightarrow a = a | b$
- $\wedge=$ , `xor_eq`, //  $a \wedge= b \Leftrightarrow a = a \wedge b$



- $\sim$ , compl : complément à 1

Table de vérité NOT

A	NOT A
0	1
1	0

- $\ll$ , décalage à gauche
- $\gg$ , décalage à droite

Exemple :  $p = n \ll 3;$

p égal n décalé de 3 bits sur la gauche



$\text{expr1} \text{ ? } \text{expr2} \text{ : } \text{expr3}$

si (expr1) alors expr2 sinon expr3

Exemple : fonction min

if (y<z) x=y; else x=z;

Equivalent à

$x = (y < z) \text{ ? } y \text{ : } z ;$

## 3 - Les bases du langage Java

### -> Les opérateurs : simplifiés



- $i=i+1$  peut s'écrire  $i++$ ;       $(++i;)$
- $i=i-1$  peut s'écrire  $i--$ ;       $(--i;)$
- $a=a+b$  peut s'écrire  $a+=b$ ;
- $a=a-b$  peut s'écrire  $a-=b$ ;
- $a=a\&b$  peut s'écrire  $a\&=b$ ;



- Quelle utilisation ?
- Opérateurs classiques
- Les expressions
- Et - Ou - !



- Tous les tests :
  - si (expression vraie) ...
  - tant que (expression vraie) ...
- Valeurs booléennes
  - Vrai (1)
  - Faux (0)

## 3 - Les bases du langage Java

### -> Les opérateurs : logiques - Opérateurs classiques



- Le 'ET'

A	B	ET
0	0	0
0	1	0
1	0	0
1	1	1

- Le 'OU'

A	B	OU
0	0	0
0	1	1
1	0	1
1	1	1





- Egalité

$a == b$

- Inégalité

$a != b$

- Relations d'ordre

$a < b$     $a <= b$     $a > b$     $a >= b$

- Expression

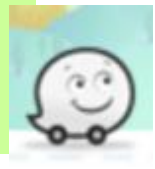
a est vraie si a est différent de 0



- Et Logique
  - expression1 **&&** expression2
  - *si expression1 et expression2 sont vraies*
- Ou Logique
  - expression1 **||** expression2
  - *si expression1 ou expression2 est vraie*
- ! (Non)
  - **!**expression
  - *si expression est fausse*

## 3 - Les bases du langage Java

### -> Les structures de contrôle et boucle



- Si ... alors ... sinon ... => `if () {} else {}`
- Au cas ... ou faire ... => `switch () { case }`
- Tant que ... faire ... => `while () {}`
- Répéter ... tant que ... => `do {} while ()`
- Boucle Pour ... faire ... => `for () {}`
- Boucle Pour chaque ... faire ... => `for () {}`
- Rupture de séquence => `break, continue, return`

### 3 - Les bases du langage Java

#### -> Les structures de contrôle et boucle

: si ... alors ... sinon ... => if () {} else {}



Si (expression conditionnelle vraie)  
alors { instructions 1 }  
sinon { instructions 2 }

```
if (expression) {  
    instructions 1;  
}  
else {  
    instructions 2;  
}
```

```
if (expression) {  
    instructions 1; /* bloc else  
    optionnel */  
}
```

```
if (expression)  
    instruction; /* bloc d'1  
    instruction */
```

```
if (expression) instruction1;  
else instruction2;
```

### 3 - Les bases du langage Java

#### -> Les structures de contrôle et boucle

: si ... alors ... sinon ... => if () {} else {} : Exemple



```
1 package helloworld;
2 import java.util.Scanner;
3
4 public class HelloWorld {
5
6
7     public static void main(String[] args) {
8         Scanner sc = new Scanner(System.in);
9         System.out.println("Hello World ( Bonjour le monde )");
10        System.out.println("Appuyer sur la touche q pour quitter ");
11        String quit = sc.nextLine();
12        if ( "q".equals(quit) ) return ;
13        if ( "Q".equals(quit) ) {
14            System.out.println("Appuyer sur la touche q et pas Q pour quitter ");
15            quit = sc.nextLine();
16        } else {
17            System.out.println("Appuyer sur la touche q pour quitter ");
18            quit = sc.nextLine();
19        }
20        if ( "q".equals(quit) ) return ;
21        else System.out.println("Dernier essai, Appuyer sur la touche q pour quitter ");
22        quit = sc.nextLine();
23        System.out.println("Good bye");
24    }
25 }
26
27
```

### 3 - Les bases du langage Java

#### -> Les structures de contrôle et boucle

: Au cas ... ou faire ...=> switch () { case }

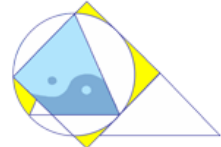


Au cas où la variable vaut :

valeur 1 : faire ... ;

valeur 2 : faire ... ; etc.

```
switch (variable de type char ou int) {  
    case valeur 1 : ...;  
        ...;  
        break;  
    case valeur 2 : ...;  
        ...;  
        break;  
    default :      ...;  
        ...;  
}
```



### 3 - Les bases du langage Java

#### -> Les structures de contrôle et boucle

: Au cas ... ou faire ...=> switch () { case }Exemple



```

7 public static void main(String[] args) {
8     Scanner sc = new Scanner(System.in);
9     System.out.println("Hello World ( Bonjour le monde )");
10    System.out.println("Appuyer sur la touche q pour quitter ");
11    String quit = sc.nextLine();
12    switch (quit.charAt(0)) {
13        case 'q':
14            return;
15        case 'Q':
16            System.out.println("Appuyer sur la touche q et pas Q pour quitter ");
17            quit = sc.nextLine();
18            break;
19        case 'r':case 'R':case 'S':case 'T':
20            System.out.println("Appuyer sur la touche q et pas " + quit + " pour quitter ");
21            quit = sc.nextLine();
22            break;
23        default:
24            System.out.println("Appuyer sur la touche q et pas " + quit + " pour quitter ");
25            quit = sc.nextLine();
26            break;
27    }
28
29    if ("q".equals(quit)) {
30        return;
31    } else {
32        System.out.println("Dernier essai, Appuyer sur la touche q pour quitter ");
33    }
34    quit = sc.nextLine();
35    System.out.println("Good bye");
36 }

```

### 3 - Les bases du langage Java

-> Les structures de contrôle et boucle

: Tant que ... faire => while    ()    {}



Tant que (expression vraie) faire  
{bloc d 'instructions}

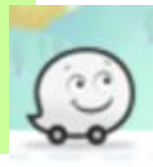
```
while (expression vraie) {  
    bloc d 'instructions }  
}
```



### 3 - Les bases du langage Java

-> Les structures de contrôle et boucle

: Tant que ... faire => while    () {} : Exemple



```
package helloworld;
```

```
import java.util.Scanner;
```

```
public class HelloWorld {
```

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

```
        Scanner sc = new Scanner(System.in);
```

```
        String quit = new String();
```

```
        while (! "q".equals(quit)) {
```

```
            System.out.println("Hello World ( Bonjour le monde )");
```

```
            System.out.println("Appuyer sur la touche q pour quitter ");
```

```
            quit = sc.nextLine();
```

```
        }
```

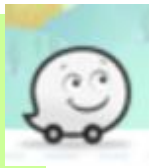
```
        System.out.println("Good bye");
```

```
    }
```

```
}
```

### 3 - Les bases du langage Java

-> Les structures de contrôle et boucle  
: Répéter ... tant que ...=> `do {} while ()`



Répéter {bloc d 'instructions}  
tant que (expression vraie)

```
do {bloc d 'instructions} while  
(expression vraie);
```

### 3 - Les bases du langage Java

-> Les structures de contrôle et boucle

: Répéter ... tant que ...=> `do {} while ()` : Exemple



```
1  package helloworld;
2
3  import java.util.Scanner;
4
5  public class HelloWorld {
6
7      public static void main(String[] args) {
8          Scanner sc = new Scanner(System.in);
9          String quit;
10         do {
11             System.out.println("Hello World ( Bonjour le monde )");
12             System.out.println("Appuyer sur la touche q pour quitter ");
13             quit = sc.nextLine();
14         }while (! "q".equals(quit));
15         System.out.println("Good bye");
16     }
17 }
```

### 3 - Les bases du langage Java

-> Les structures de contrôle et boucle

: Boucle Pour ... faire ... =>

for () {}



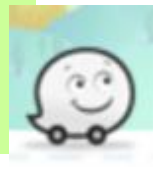
Pour (initialisation;condition;modification) faire {  
    bloc d 'instructions}

```
for (initialisation;condition;modification) {  
    bloc d 'instructions}
```

### 3 - Les bases du langage Java

-> Les structures de contrôle et boucle

: Boucle Pour ... faire ... => for () {} : Exemple



```
1  package helloworld;
2
3  import java.util.Scanner;
4
5  public class HelloWorld {
6
7      public static void main(String[] args) {
8          Scanner sc = new Scanner(System.in);
9          String quit="";
10
11          for (; !"q".equals(quit);) {
12              for (int i = 0; i < 5; i++) {
13                  System.out.println("Hello World ( Bonjour le monde )");
14              }
15              System.out.println("Appuyer sur la touche q pour quitter ");
16              quit = sc.nextLine();
17          }
18
19          System.out.println("Good bye");
20      }
21  }
22
```



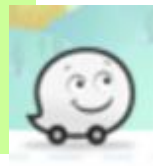
Pour (déclaration: expression) faire {  
    bloc d 'instructions }





```
for (déclaration: expression) {  
    bloc d 'instructions}
```

### 3 - Les bases du langage Java

#### -> Les structures de contrôle et boucle

: Boucle Pour chaque ... faire ... => for () {} : Exemple



```
1  package helloworld;
2
3    import java.util.Scanner;
4
5  public class HelloWorld {
6
7       public static void main(String[] args) {
8          System.out.println("Good bye");
9          int v[] = {0, 1, 2, 3, 4, 5};
10         for (int i : v) {
11             System.out.println("valeur " + i);
12         }
13         System.out.println();
14         String[] data = {"Nice", "Marseille"};
15          for (String s : data) {
16             System.out.println(s);
17         }
18     }
19 }
```



- ***Return[valeur]*** : permet de quitter immédiatement la fonction en cours.
- ***break*** : permet de passer à l'instruction suivant l'instruction *while*, *do*, *for* ou *switch* la plus imbriquée.
- ***continue*** : saute directement à la dernière ligne de l'instruction *while*, *do* ou *for* la plus imbriquée.



### 3 - Les bases du langage Java

#### -> Les structures de contrôle et boucle

#### : Rupture de séquence - break, continue, return : Exemple



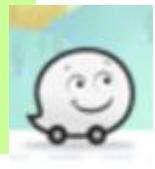
```
package helloworld;

import java.util.Scanner;

public class HelloWorld {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String quit = "";
        while (!"q".equals(quit)) {
            for (int i = 0; i < 5; i++) {
                System.out.println("Hello World ( Bonjour le monde ) : num => " + i);
                System.out.println("Appuyer sur la touche 'o' pour quitter la boucle ");
                quit = sc.nextLine();
                if ("o".equals(quit)) {
                    break;
                }
            }
            System.out.println(" Appuyer sur la touche 'r' pour recommencer la boucle");
            System.out.println(" Appuyer sur la touche 'q' pour quitter ");
            quit = sc.nextLine();

            if ("r".equals(quit)) {
                continue;
            }
            System.out.println("Suite => ");
            if (!"q".equals(quit)) {
                return;
            }
        }
        System.out.println("Good bye");
    }
}
```



`String mes =" Ceci est " + " une chaine ";`

- Les chaînes peuvent être concaténées à des nombres :

```
int ans = 10;
```

```
String mes=" Je suis age de " + ans + " ans ";
```

- Les chaînes peuvent être concaténées à tout type d'objet :

```
Personne durand = new Personne();
```

```
String mes=" Je suis une personne " + durand;
```

sera compilé comme

```
String mes=" Je suis une personne " + durand.toString();
```



String salut = "Hello";

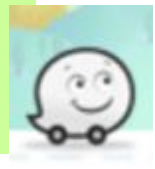
- égalité : méthode equals

```
if (salut.equals("Hello") ) // vrai
```

```
If ("Hello".equals(salut) )
```

- sous-chaînes : méthode substring

```
if (salut.substring(0,4).equals("Hell")) // vrai
```



```
int [] a ; // tableaux d 'entiers  
double [] [] m ; // matrice de double  
String [] jours= {"Lundi","mardi", ... "Dimanche"};
```

- **Construction du tableau par new**

```
a=new int[10] ;  
m=new double [3][5] ; // 3 lignes et 5 colonnes
```

- **Utilisation**

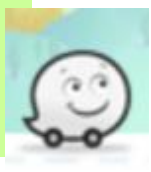
```
m[i][j]=x;
```

- **Taille**

```
a.length // = 10
```

## 3 - Les bases du langage Java

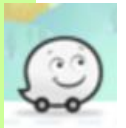
### -> les tableaux : initialisation explicite d'un tableau



```
//int tableau5[5] = {10,20,30,40,50};  
//int tableau6[3][2] = {{5,1},{6,2},{7,3}};  
  
int [] tableauInt3 = {10,20,30,40,50};  
int tableauInt4[][] = {{5,1},{6,2},{7,3}};  
double tableauDouble1[][] = {{5.,1.,3.},  
                                {6.0,2.},  
                                {1.,2.,3.,4.,7,3}};
```

## 3 - Les bases du langage Java

### -> les tableaux : La déclaration des tableaux



```
int tableauInt1[] = new int[10]; /* <=>  int[] tableauInt1 = new int[20]; */  
|  
int tableauInt2[]; // déclaration  
tableauInt2 = new int[30]; //allocation  
  
float tableauFloatMulti[][] = new float[10][10];
```

## 3 - Les bases du langage Java

### -> les tableaux : Le parcours d'un tableau



```
int [] tableauInt = {10,20,30,40,50};

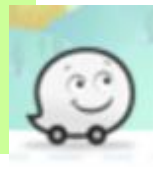
for (int i = 0; i < tableauInt.length ; i++) {
    System.out.println( "T["+i+"]=" + tableauInt[i] );
}

System.out.println("-----");
int k=0;
for (int val: tableauInt) {
    System.out.println( "T["+k+"]=" + val );
    ++k;
}
```

Remarque : un tableau qui dépasse sa capacité, lève une exception du type `java.lang.arrayIndexOutOfBoundsException`.

## 3 - Les bases du langage Java

### -> Les conversions de types



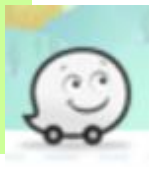
Classe	Rôle
String	pour les chaînes de caractères Unicode
Integer	pour les valeurs entières (integer)
Long	pour les entiers longs signés (long)
Float	pour les nombres à virgule flottante (float)
Double	pour les nombres à virgule flottante en double précision (double)

Remarque : La conversion peut entraîner une perte d'informations



## 3 - Les bases du langage Java

### -> Les conversions de types

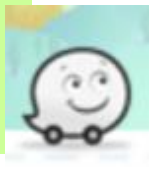


- La conversion d'un entier int en float

```
int entier = 5;  
float flottant = (float) entier;
```

- La conversion d'un entier int en entier long

```
int entier = 5;  
Integer nombre= new Integer(entier);  
long f = nombre.intValue();
```



- La conversion d'un entier int en chaîne de caractères String

```
int i = 10;  
String texte = new String();  
texte = texte.valueOf(i);
```

- La conversion d'une chaîne de caractères String en entier int

```
String texte = new String(" 10 ");  
Integer monnombre = new Integer(texte);  
int i = monnombre.intValue();
```



- Un caractère

```
char uncaractere = 'a';
```

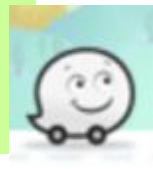
- Une chaine de caractère

```
String texte = "Bonjour ";
```

### 3 - Les bases du langage Java

#### -> La manipulation des chaînes de caractères

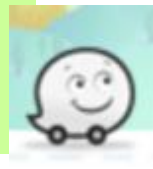
#### : Caractères spéciaux



Caractères spéciaux	Affichage
\'	Apostrophe
\"	Guillemet
\\	Antislash
\t	Tabulation
\b	Retour arrière
\r	Retour chariot
\f	Saut de page
\n	Saut de ligne
\0ddd	Caractère ASCII ddd (octal)
\xdd	Caractère ASCII dd (hexadécimal)
\udddd	Caractère Unicode dddd (hexadécimal)

## 3 - Les bases du langage Java

### -> La manipulation des chaînes de caractères : opération divers



- Addition de chaine
- Comparaison de chaine
- Longueur d'une chaine
- Modification de la casse d'une chaine

```
String texte1 = " texte";  
texte1 += " texte" ;  
texte1 += " 2 | ";
```

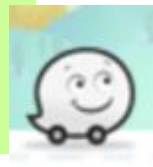
```
String texte2 = " texte 2 " ;  
if (texte1.equals(texte2)) {  
    System.out.println("OK");  
} else {  
    System.out.println("KO");  
}
```

```
System.out.println(texte1.length());
```

```
String textemin = " texte " ;  
String textemaj = textemin.toUpperCase();
```

# 3 - Les bases du langage Java

## -> La manipulation des chaînes de caractères



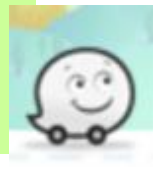
<https://docs.oracle.com/javase/8/docs/api/index.html?java/lang/String.html>



The screenshot shows the Oracle Java API documentation for the `String` class. The left sidebar lists various Java packages and classes. The main content area displays the `String` class details, including its inheritance hierarchy (extending `Object` and implementing `Serializable`, `Comparable<String>`, and `CharSequence`). It explains that `String` represents character strings and that all string literals in Java programs are implemented as instances of this class. The documentation also shows examples of how strings can be used, such as creating a `String` object and using methods like `println`, `substring`, and `toUpperCase`. The bottom of the page lists related classes and methods, including `Object.toString()`, `StringBuffer`, `StringBuilder`, `Charset`, and `Serialized Form`.

Liens Divers:

- [http://www.tutorialspoint.com/java/java\\_strings.htm](http://www.tutorialspoint.com/java/java_strings.htm)
- <http://www.java-examples.com/java-string-examples>



# Références

- **L'accès aux objets et tableaux se fait par référence**  
    <nom-de-classe> <nom-objet>;
- **Le constructeur " new " alloue la mémoire et initialise les données de l'objet**

Personne Durand = new Personne(" Durand ", 20);

☞ String nom=" Durand "   ou   nom=new String(" Durand ");

int array[][] ;

array=new int[4][];

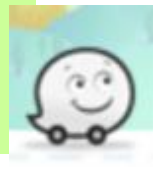
ou   array =new int[4][3];

for (int i=0; i<4; i++)

    array=new int[3];

## 3 - Les bases du langage Java

### -> Passage par valeur ou par référence

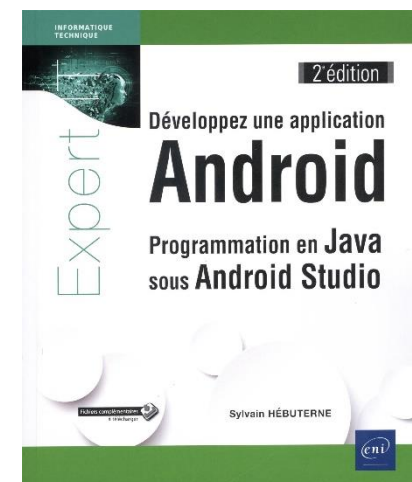
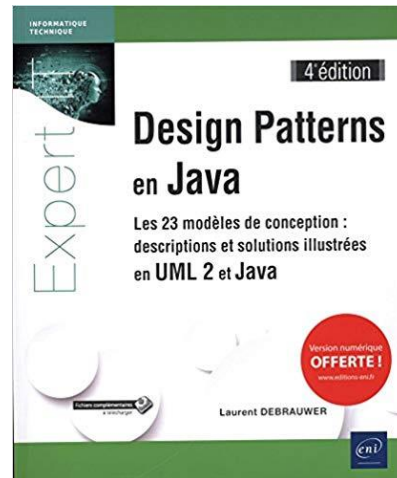
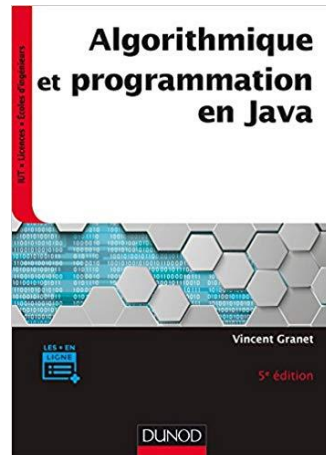
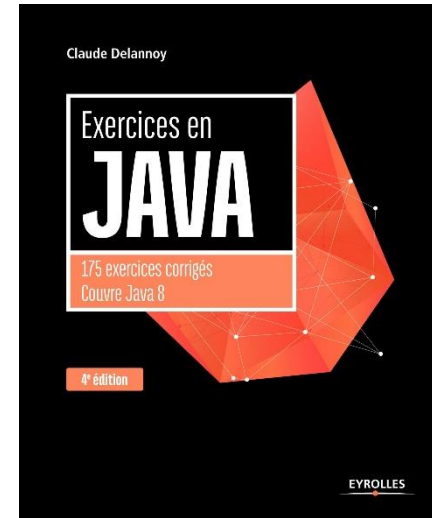
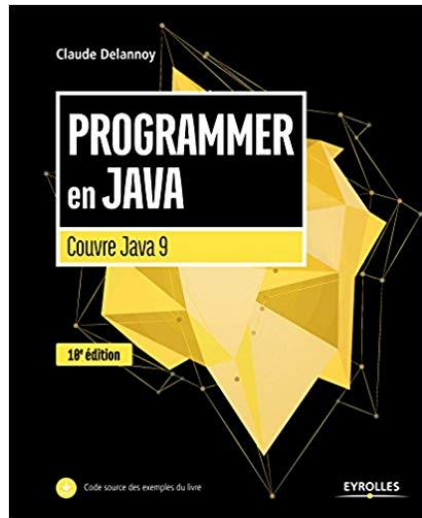


Les arguments aux méthodes Java sont passés :

- **par valeur** pour les types de base  
c'est une copie qui est passée à la méthode
- **par référence** pour les objets et tableaux



## 4 - Bibliographies



## 4 - Bibliographies

### Java :

- <http://gaetan.dussaux.free.fr/cours/java/12.htm#autres>
- <http://www.jmdoudoux.fr/java/dej/indexavecframes.htm>
- <http://www.tutorialspoint.com/java/>

### IntelliJ Idea :

[https://www.tutorialspoint.com/intellij\\_idea/index.htm](https://www.tutorialspoint.com/intellij_idea/index.htm)

### Netbeans :

- <http://www.bestcours.com/cours-pdf-netbeans-java>

### Android :

- <http://www.univ-orleans.fr/lifo/Members/Jean-Francois.Lalande/enseignement/android/cours-android.pdf>
- <https://olegoaer.developpez.com/cours/mobile/>
- <https://perso.univ-rennes1.fr/pierre.nerzic/Android/poly.pdf>

## 4 - Bibliographies

<https://objis.com/tutoriel-java-n12-acces-base-de-donnees-mysql/>

<https://www.jmdoudoux.fr/java/dej/chap-jdbc.htm>