

Variables (types, constantes, cast et String)

Les types primitifs

```
/* Code à intégrer dans la fonction main d'une classe */

// On peut déclarer une variable avant de l'initier
int entier;
entier = 42;

// Ou on peut faire les deux en même temps
boolean test = true;
char lettre = 'A'; // Attention, le ' est nécessaire et ne peut pas être remplacé par "
byte octet = 127
double nombreAVirgule = 5.2;

// On peut aussi créer une constante, sa valeur ne pourra pas changer
final int reponseALaVie = 42;
```

Il y a 8 types primitifs en Java:

Type	Taille en bits	Domaine de valeurs	Information supplémentaire
<code>boolean</code>	?	0 (false), 1 (true)	La taille dépend de la JVM
<code>char</code>	16	N'importe quel caractère unicode	N/A
<code>byte</code>	8	\$ -128 \$ → \$ +127 \$	N/A
<code>short</code>	16	\$ -2 ¹⁵ \$ → \$ 2 ¹⁵ - 1 \$	N/A
<code>int</code>	32	\$ -2 ³¹ \$ → \$ 2 ³¹ - 1 \$	N/A
<code>long</code>	64	\$ -2 ⁶³ \$ → \$ 2 ⁶³ - 1 \$	N/A

Type	Taille en bits	Domaine de valeurs	Information supplémentaire
float	16	\$ -3.4 * 10 ^{38} \$ → \$ 3.4 * 10 ^{38} \$	Nombre à virgule (virgule flottante). La précision est de \$ 1.4 * 10 ^{-45} \$
double	64	\$ -1.7 * 10 ^{308} \$ → \$ 1.7 * 10 ^{308} \$	Comme float mais avec une précision de \$ 4.9 * 10 ^{-324} \$

Convertir entre les types avec cast

```
// Conversion de float en int avec cast
double x = 42.5;
int y = (int)x; // 42
int z = (int)42.5; // 42
int a = (int)(42.0 + 0.5); // 42
```

On peut aussi transformer une valeur en une autre en utilisant un *cast*. Comme vu ci dessus en précisant le type cible en ().

A savoir qu'ici, dans le cas d'une conversion d'un nombre en virgule flottante en entier, on perd les informations des décimales. Cast va toujours arrondir vers le bas dans une conversion comme celle ci. Pour avoir plus de controle sur l'arrondis, on peut utiliser la classe java Math (voir plus tard).

Introduction aux Strings (chaîne de caractères)

```
String maChaine = "Hello, World!";
System.out.println(maChaine);

// Convertir un string en un int (cast est impossible car String est une classe)
String nombreStr = "42"; // "42"
int nombreInt = Integer.parseInt(nombreStr); // 42
```

String n'est pas un type primitif, mais bien une classe (ce qui est pourquoi la première lettre est en majuscule) qui correspond à une chaîne de caractère, dans un chapitre suivant nous allons voir quelques méthodes qui peuvent être appliquées aux objets de la classe "String".

N'étant pas un type primitif nous ne pouvons pas utiliser cast dessus mais on peut utiliser d'autres méthodes d'autres classes comme `Integer.parseInt()`, il existe aussi d'autres méthodes du même

genre comme `Double.parseDouble()`.

En savoir plus

- [Wikiversity FR - Java: Variables et types](#)
- [Oracle Docs - String](#)
- [Oracle Docs - Integer.parseInt](#)

Revision #1

Created 27 April 2023 06:31:19 by SnowCode

Updated 27 April 2023 06:35:12 by SnowCode