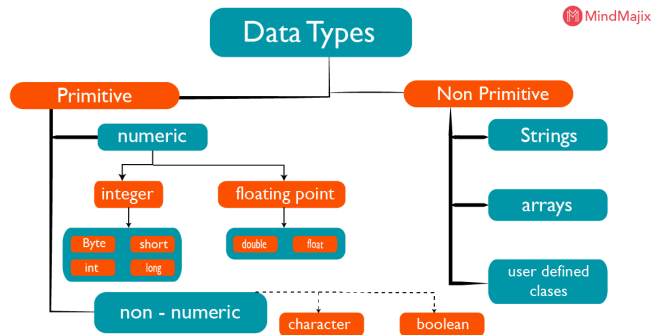


# Les Types de Variables en Java



En Java, il existe plusieurs types de variables, chacune ayant des caractéristiques spécifiques. Les variables peuvent être classées selon plusieurs critères : par leur type de données (primitif ou référence) et par leur portée (locale, d'instance, ou de classe).

## 1. Types de données primitifs

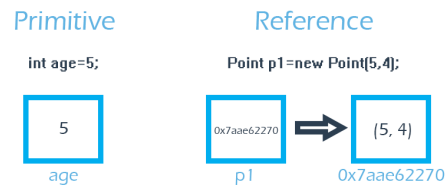
TYPE	DESCRIPTION	DEFAULT	SIZE	EXAMPLE LITERALS	RANGE OF VALUES
boolean	true or false	false	1 bit	true, false	true, false
byte	twos complement integer	0	8 bits	(none)	-128 to 127
char	unicode character	������	16 bits	'�', ������', ����', ��, '�', '��', '�', �'	character representation of ASCII values 0 to 255
short	twos complement integer	0	16 bits	(none)	-32,768 to 32,767
int	twos complement integer	0	32 bits	-2, -1, 0, 1, 2	-2,147,483,648 to 2,147,483,647
long	twos complement integer	0	64 bits	-2L, -1L, 0L, 1L, 2L	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
float	IEEE 754 floating point	0.0	32 bits	1.23e100f, -1.23e-100f, .3f, 3.14F	upto 7 decimal digits
double	IEEE 754 floating point	0.0	64 bits	1.23456e300d, -1.23456e-300d, 1e1d	upto 16 decimal digits

Les variables de type primitif contiennent directement des valeurs simples et ne sont pas des objets. Java dispose de huit types de données primitifs :

- **byte** : Entier de 8 bits, plage de valeurs de -128 à 127.
- **short** : Entier de 16 bits, plage de valeurs de -32 768 à 32 767.
- **int** : Entier de 32 bits, plage de valeurs de  $-2^{31}$  à  $2^{31}-1$ .
- **long** : Entier de 64 bits, plage de valeurs de  $-2^{63}$  à  $2^{63}-1$ .

- **float** : Nombre à virgule flottante de 32 bits.
- **double** : Nombre à virgule flottante de 64 bits.
- **char** : Caractère Unicode de 16 bits (représente un seul caractère, comme 'a' ou 'A').
- **boolean** : Peut avoir deux valeurs, **true** ou **false**.

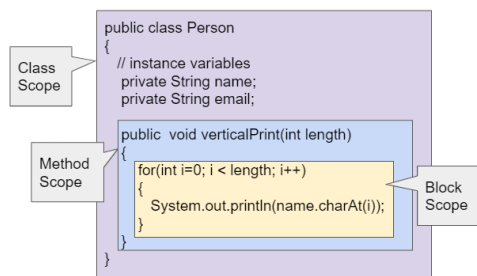
## 2. Types de données de référence



Les variables de type référence contiennent des références à des objets (instances de classes). Contrairement aux types primitifs, ces variables ne contiennent pas directement les valeurs, mais plutôt des pointeurs vers des objets dans la mémoire.

- **Classes** : Les variables qui sont des instances de classes (**String**, **Array**, **CustomClass**, etc.).
- **Interfaces** : Les variables peuvent être de type interface et référencer des objets qui implémentent ces interfaces.
- **Tableaux** : En Java, les tableaux sont des objets, donc une variable de tableau est une référence à un objet tableau.

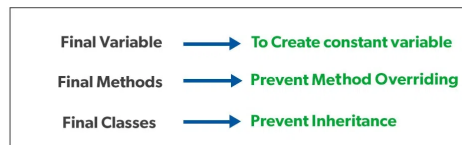
## 3. Portée des variables



Les variables en Java peuvent également être classées en fonction de leur portée :

- **Variables locales** : Déclarées dans une méthode, un constructeur ou un bloc, et ne sont accessibles qu'à l'intérieur de cette méthode, ce constructeur ou ce bloc.
- **Variables d'instance (ou de champ)** : Déclarées dans une classe, mais en dehors des méthodes, des constructeurs ou des blocs. Elles sont associées aux objets (une copie par instance).
- **Variables de classe (ou statiques)** : Déclarées avec le mot-clé **static** dans une classe. Elles sont partagées entre toutes les instances de la classe (une seule copie partagée par toutes les instances).

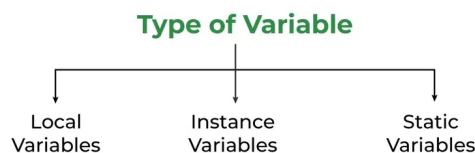
#### 4. Constantes



Les variables peuvent être déclarées avec le mot-clé **final**, ce qui signifie que leur valeur ne peut pas être modifiée après leur initialisation. Ce sont des constantes.

- **Variables final** : Utilisées pour définir des constantes. Elles doivent être initialisées lors de leur déclaration ou dans le constructeur.

Exemple en code Java :



```

public class Exemple {
    // Variable de classe (statique)
    static int compteur;

    // Variable d'instance
  
```

```
int numero;

public void methode() {
    // Variable locale
    int somme = 0;

    // Utilisation d'une constante
    final int CONSTANCE = 10;

    // Exemple d'utilisation
    somme = CONSTANCE + numero;
}
}
```

## Conclusion

En résumé, Java propose différents types de variables en fonction de la nature des données qu'elles doivent stocker et de la portée dans laquelle elles sont définies.