

# Le Langage Java

1<sup>re</sup> année

J. Beleho (bej)   C. Leruste (clr)   M. Codutti (mcd)  
P. Bettens (pbt)   F. Servais (srv)   C. Leignel (clg)  
D. Nabet (dna)   J. Lechien (jlc)

Haute École de Bruxelles — École Supérieure d'Informatique

Année académique 2014 / 2015

## Séance 9

### Le survol des tableaux

- Les tableaux (survol)
- Erreurs fréquentes

# Avertissement

Nous présentons ici une vue **simplifiée**  
des tableaux en Java afin de **coller**  
à votre cours d'**algorithmique**.

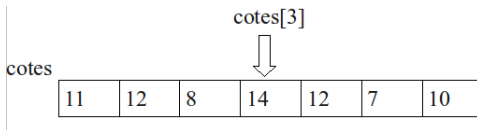
Nous aurons l'occasion d'être plus précis  
lors d'une prochaine leçon.



tableau  
plusieurs variables de même type

# Présentation

Nécessité de manipuler **plusieurs variables similaires** auxquelles on accède par un **indice**



Pourquoi pas plusieurs variables ?

# Présentation

Écriture **compacte** et qui s'**adapte** à la taille

## Exemple

En logique, si **tab** est un tableau de  $N$  entiers

```
pour i de 1 à N faire  
    écrire tab[i]  
fin pour
```



# Déclaration

Type[] identifier

# Déclaration

## Exemples

- ▶ **int []** est le type *tableau d'entiers*
- ▶ **String []** est le type *tableau de chaines de caractères*

```
int [] cotes;  
String [] noms;
```



new

Création

```
identifiant = new Type[taille]
```

# Création

## Exemple

- ▶ `new int[3]`
- ▶ `new String[taille]` où `taille` est défini

```
int [] entiers ;  
entiers = new int[3];
```

## Remarque

La déclaration et la création peuvent être combinées

```
int [] entiers = new int[3];
```



# Initialisation

```
identifier = new Type[] {x, x}
```

# Initialisation

## Exemple

- ▶ `new int[] {42, 17, -5}`
- ▶ `new String[] {"foo", "bar"}`

```
int [] entiers = new int[] {0x2A, 021, -5};  
String [] noms = new String[]  
    {"Victoria", "Melanie", "Melanie", "Emma", "Geri"}
```

## Remarque


Par défaut, les éléments sont initialisés à **0** (numériques)  
ou **false** (booléens)

# Création et initialisation

## Cas particulier

Création du tableau et initialisation en une seule étape, en **donnant ses valeurs**

```
int[] entiers = {0x2A, 021, -5};  
double[] pseudoRéels = {4.5, 1E-4, -4.12, Math.PI}
```



Accès aux éléments

[i]

# Accès aux éléments

[i]

- ▶ 0 est l'indice de départ
- ▶ indices de **0** à **taille du tableau - 1**
- ▶ la taille du tableau est son nombre d'éléments

```
int[] entiers = {3,14,15};  
int entier = entiers[2]; // entier vaut 15  
entiers[1] = 85;  
entier = entiers[entier+1]; //entier vaut 85
```

# Accès aux éléments

## Exemple

```
package be.heb.esi.lg1.tutorials.tableaux;

public class InitialisationTableau {
    public static void main(String[] args) {
        int[] entiers = new int[10];
        for(int i = 0; i < 10; i++) {
            entiers[i] = i;
        }
    }
}
```





Un tableau connaît  
**sa taille**

`identifier.length`

# Taille

## Exemple

```
int[] entiers = {4,5,6};  
int taille = entiers.length;  
System.out.println ( taille ); // écrit 3
```

# Taille

## Exemple

```
package be.heb.esi.lg1.tutorials.tableaux;

public class SimpleParcoursAscendant {
    public static void main(String[] args){
        int[] entiers = {1,2,3,4,5,6,7,8,9,10};
        for(int i = 0; i < entiers.length; i = i + 1) {
            System.out.println ( entiers [ i ] );
        }
    }
}
```

# Taille

## Exemple

```
package be.heb.esi.lg1.tutorials.tableaux;

public class SimpleParcoursDescendant {
    public static void main(String[] args){
        int[] entiers = {1,2,3,4,5,6,7,8,9,10};
        for(int i = entiers.length - 1; i >= 0; i = i-1) {
            System.out.println ( entiers [ i ] );
        }
    }
}
```

# Tableau et méthode

Un tableau peut être un paramètre d'une méthode.

**Exemple** : Afficher un tableau

```
public static void afficher ( int[] entiers ) {  
    for(int i = 0; i<entiers.length; i++) {  
        System.out. println ( entiers [ i ] );  
    }  
}
```

► L'appel pourrait être

```
int[] cotes = {12,8,10,14,9};  
afficher ( cotes );
```

# Tableau et méthode

En Java, **passage de paramètre par valeur**.

Pour un tableau, cela signifie que l'on ne peut pas modifier le tableau dans son ensemble mais que l'on pourra modifier ses éléments.

# Tableau et méthode

## Exemple

```
public static void remplir( int[] entiers , int val ) {  
    for(int i = 0; i < entiers.length; i++) {  
        entiers[i] = val;  
    }  
}
```

- L'appel pourrait être

```
int[] cotes = new int[16]; // Ne pas oublier de le créer  
remplir( cotes, 20 );
```

# Tableau et méthode

## Exemple

```
public static void methodeFausse( double[] réels) {  
    double[] réelsDePassage = {4.2, -7, Math.PI};  
    réels = réelsDePassage; // INUTILE  
}
```

Quel que soit l'appel, le tableau que l'on passe en paramètre ne sera pas modifié



# Tableau et méthode

Un tableau peut être une valeur de retour

**Exemple** : Créer un tableau avec valeur

```
public static int[] créer( int taille , int val ) {  
    int[] entiers = new int[ taille ];  
    for(int i = 0; i < taille ; i++) {  
        entiers [ i ] = val;  
    }  
    return entiers ;  
}
```

► L'appel pourrait être

```
int[] cotes = créer(16, 20);
```

# Erreurs fréquentes

## Exceptions

- ▶ `NullPointerException` : si vous essayez d'accéder à un élément d'un tableau qui n'a pas été créé (le tableau vaut **null** dans ce cas)
- ▶ `ArrayIndexOutOfBoundsException` : si vous donnez un indice qui n'existe pas (ex : `tab[10]` quand il n'y a que 10 éléments dans le tableau)

# Crédits

Ces slides sont le support pour la présentation orale de l'unité d'enseignement **DEV1-JAV** à HEB-ÉSI

## Crédits

Les distributions Ubuntu et/ou debian  
du système d'exploitation **GNU Linux**.

**LaTeX/Beamer** comme système d'édition.

**Git** et GitHub pour la gestion des versions et le suivi.

**GNU make**, **rubber**, **pdfnup**, ... pour les petites tâches.

## Images et icônes

deviantart, flickr, The Noun Project

