

# 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.P. Bishop (bis)   S. Drobisz (sdr)

Haute École de Bruxelles-Brabant — École Supérieure d'Informatique

Année académique 2016 / 2017

## Séance 3

### Survol module et structures séquentielles

- Code modulaire (premier survol)
- Algorithmes
- séquentiels (survol)
- Constantes
- Conventions

# Les variables

## Les types disponibles

En Algorithmique	En Java
Entier	<b>int</b>
Réel	<b>double</b>
Chaine	<b>char</b> , <b>String</b>
Booléen	<b>boolean</b>

# Les calculs

Opérateurs :

$+$   $-$   $*$   $/$   $\%$

# Variable

On peut stocker une valeur, le résultat d'un calcul dans une **variable**

## Déclaration

```
typeVariable nomVariable;
```

► Exemples :

```
int nb1;  
String nom;
```

# Variable

## Assignment

Via le symbole =

► Exemples :

```
nb1 = 1;  
nb1 = 3+4;  
nom = "Java";  
moyenne = (nombre1 + nombre2) / 2.0;  
dé = Math.random() * 6 + 1
```



Découper le code

Crédit photo

**Appel** de méthode



# Appel d'une méthode

Une méthode est une **boîte noire**



Pour l'utiliser, il faut connaître :

- ▶ son nom ;
- ▶ ce dont elle a besoin ;
- ▶ ce qu'elle retourne ;
- ▶ mais **pas comment** elle fait ;

Pas de **comment** ?  
Un simple **quoi**



# Appel d'une méthode

À partir du code d'une **autre classe**

- ▶ `NomClasse.nomMéthode(...)`
- ▶ **Exemples :**

```
double racine = Math.sqrt(4.0);  
double aléatoire = Math.random();  
int nb = -10;  
int absolu = Math.abs(nb);
```

*Un appel de méthode au sein de la classe ne nécessite pas de répéter le nom de la classe*

**Définition** d'une méthode

# Définition d'une méthode

## Syntaxe

```
public static <typeRetour> <nomMéthode> ([paramètre, param...]) {  
  // instructions  
  <return résultat>;  
}
```

# Définition d'une méthode

**Exemple** : la moyenne de 2 réels

```
/* Méthode qui calcule la moyenne de 2 nombres  
* Données :  
*     nb1 : le premier nombre (un réel)  
*     nb2 : le second nombre (un réel)  
* Résultat : un réel qui représente la moyenne des 2 nombres  
*/  
  
public static double moyenne ( double nb1, double nb2 ) {  
    return (nb1 + nb2) / 2.0;  
}
```

- Appel possible (si dans la même classe)

```
System.out.println ( moyenne(34345, -3213213) );
```

# Définition d'une méthode

Si la méthode ne retourne rien

- ▶ typeRetour : `void`
- ▶ pas de **return**

# Définition d'une méthode

## Exemple :

```
public static void présenter (String nomPgm) {  
    System.out.println ("Programme_ "+nomPgm);  
}
```

- Exemple d'appel dans la même classe

```
présenter ("moyenne_de_2_nombres");
```



# Exemple

\$cat Moyenne.java

```
public class Moyenne {  
  
    public static void présenter (String nomPgm) {  
        System.out.println ("Programme_ "+nomPgm);  
    }  
  
    public static double moyenne(double nombre1, double nombre2) {  
        return (nombre1 + nombre2) / 2;  
    }  
  
    public static void main(String[] args) {  
        présenter ("qui calcule la moyenne de 2 nombres");  
        System.out.println ( moyenne(34345, -3213213) );  
    }  
}
```

## Exemple

Calculer la vitesse (en km/h) d'un véhicule dont on donne la durée du parcours (en secondes) et la distance parcourue (en mètres).

# Définition d'une méthode

```
public static double vitesseKMH ( double distanceM, double duréeS ) {  
    double distanceKM;  
    double duréeH;  
    distanceKM = distanceM/1000;  
    duréeH = duréeS/3600;  
    return distanceKM/duréeH;  
}
```

## ► Exemples d'appels dans la même classe

```
double vitesse = vitesseKMH(1000, 3600);  
double vitesseLumière = vitesseKMH(300000000, 1);  
double vitesseSon = vitesseKMH(340, 1);
```

# Lire au clavier

*Les applications modernes préfèrent les lectures dans des champs de saisies.*

Dans une console, voici une manière de faire

## Exemple

```
import java.util.Scanner;  
// ...  
Scanner clavier = new Scanner(System.in);  
// ...  
nombre1 = clavier.nextDouble();
```

# Lire au clavier - Exemple

\$cat Test.java

```
import java.util.Scanner;

public class Test {
    public static double moyenne(double nombre1, double nombre2) {
        return (nombre1 + nombre2) / 2;
    }

    public static void main(String[] args) {
        Scanner clavier = new Scanner(System.in);
        double nombre1, nombre2;
        double moyenne;
        nombre1 = clavier.nextDouble();
        nombre2 = clavier.nextDouble();
        moyenne = moyenne(nombre1, nombre2);
        System.out.println (moyenne);
    }
}
```

# Lire au clavier

Pour lire...	on écrit...
un entier	<code>nextInt()</code>
un réel	<code>nextDouble()</code>
un booléen	<code>nextBoolean()</code>
un mot	<code>next()</code>
une ligne	<code>nextLine()</code>
un caractère	<code>next().charAt(0)</code>

# Définition d'une méthode de lecture

## Exemple :

```
public static int lireEntier () {  
    Scanner clavier = new Scanner(System.in);  
    int nb;  
    System.out.println ("Entrez un nombre entier!");  
    nb = clavier.nextInt ();  
    return nb;  
}
```

- Exemple d'appel dans la même classe

```
int nb = lireEntier ();
```

# Constante locale

Une **constante** s'écrit grâce à **final**

## Exemple

```
final int X = 1;  
final int Y;  
Y = 2*X;  
X = 2; // Erreur : possède déjà une valeur  
Y = 3; // Idem
```



Handwritten text in a cursive script, likely a historical form of Hebrew or Aramaic, arranged in four rows. Each row consists of a main body of text, a smaller column of text to the right, and a single character 'I' further to the right.

## Conventions d'écriture

Printed text in a bold, blocky font, likely a historical form of Hebrew or Aramaic, arranged in one row. It consists of a main body of text, a smaller column of text to the right, and a single character 'I' further to the right.

Handwritten text in a cursive script, likely a historical form of Hebrew or Aramaic, arranged in two rows. Each row consists of a main body of text, a smaller column of text to the right, and a single character 'I' further to the right.

Handwritten text in a cursive script, likely a historical form of Hebrew or Aramaic, arranged in one row. It consists of a main body of text, a smaller column of text to the right, and a single character 'I' further to the right.

# Conventions de noms

Pour une variable :

- ▶ Tout mettre en minuscules
- ▶ Sauf les débuts de noms composés en majuscule

Pour une constante :

- ▶ Tout mettre en majuscules
- ▶ Utiliser `_` pour séparer les mots

Dans tous les cas : être explicite

# Commentaire d'une méthode

# Commentaire d'une méthode

## Documentation de `Math.abs`

### **abs**

```
public static int abs(int a)
```

Returns the absolute value of an int value. If the argument is not negative, the argument is returned. If the argument is negative, the negation of the argument is returned.

Note that if the argument is equal to the value of `Integer.MIN_VALUE`, the most negative representable int value, the result is that same value, which is negative.

#### **Parameters:**

a - the argument whose absolute value is to be determined

#### **Returns:**

the absolute value of the argument.

# Crédits

Ces slides sont le support pour la présentation orale de l'unité d'enseignement **DEV1-JAV** à HE2B-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 

