

Language Java



Fabio Cartolano

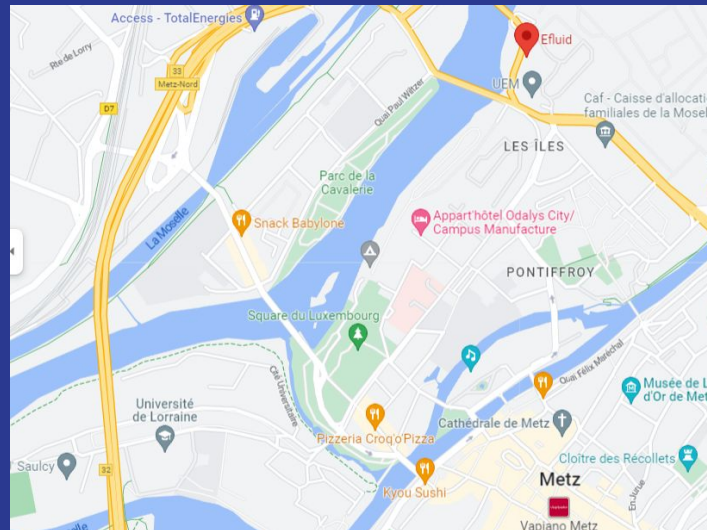
Fabio Cartolano

- DUT info
- Prépa ATS (année passerelle)
- Télécom Nancy



Efluid

- Editeur de logiciel
- Solution complète de gestion clientèle pour les fournisseurs d'énergie
- Leader sur le marché
- 40 entreprises majeures
- 30 millions de clients au quotidien.
- Plus d'1 million de lignes de code
- Majoritairement développé en Java

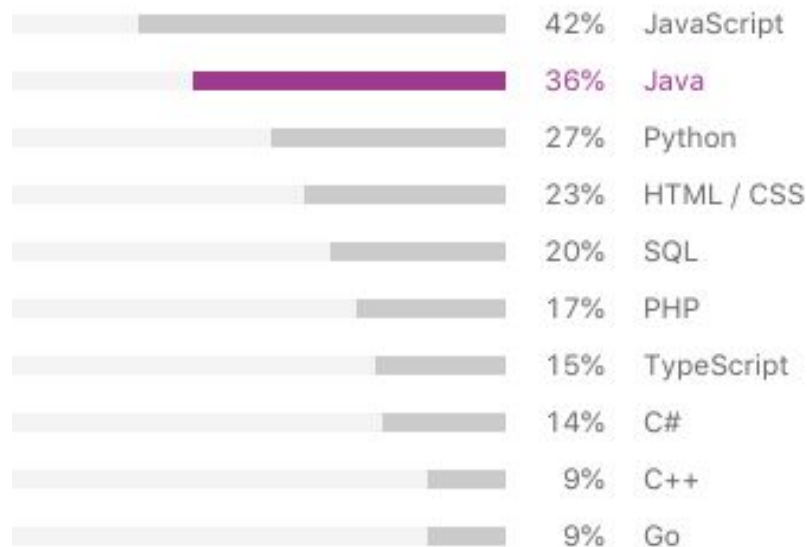




- Né au début des années 90
- Créé dans le but d'être utilisé sur les appareils grand public
- Langage compilé orienté objet
- Syntaxe claire et concise
- Gestion de mémoire automatique
- Un des langages les plus utilisés au monde

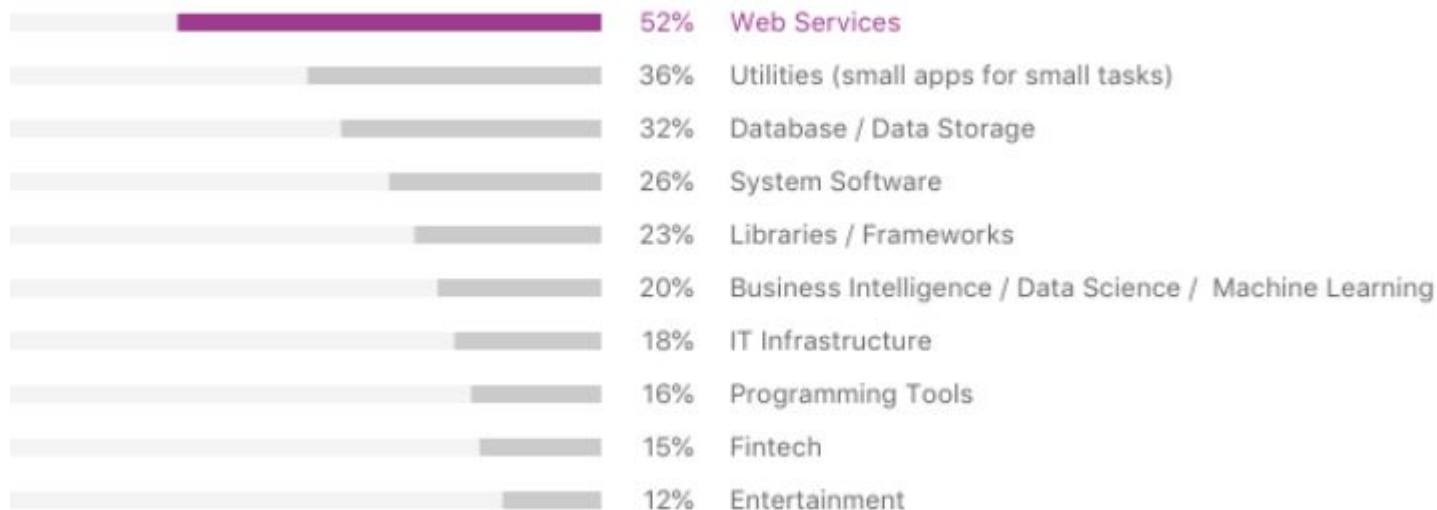
Utilisation des langages de programmation dans le milieu professionnels (2020)

Sources : Developpez.com



Types de logiciels développés avec Java (2020)

Sources : Developpez.com





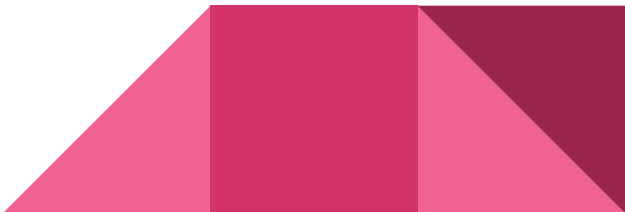
Écrire un programme en java

Définir une classe

Tous les programmes java commencent par la définition d'une classe.

Le nom de la classe et le nom du fichier doivent être les mêmes.

```
class HelloWorld {  
    ... ..  
}
```

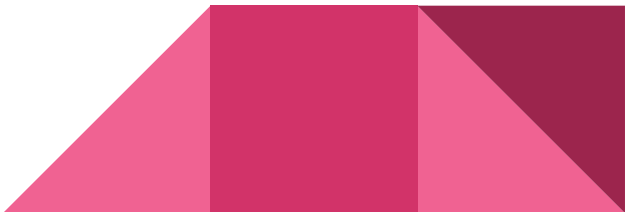


Définir la méthode “main”

Chaque application Java doit contenir une méthode “main” qui est le point de démarrage de n’importe quel programme Java.

Le compilateur Java commence toujours par exécuter le code de cette méthode.

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



Écrire dans le terminal

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



Exemple complet

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





Un langage typé

Les différents types

- Les types numériques : byte, short, int, long, float, double
- Le type booléen : boolean
- Le type caractère : char
- Les types de base : Object, String, Class
- Les types collection : List, Set, Map, etc.



Déclaration de variable

```
int age = 30;  
String nom = "Jean";  
boolean estEtudiant = true;
```

Déclaration d'une constante

```
final int NOMBRE_DE_JOUEURS = 11;  
final double PI = 3.14159;  
final String MESSAGE = "Bonjour";
```

Les structures de contrôle

Les opérations de comparaison

- Égal : $==$
- Différent : $!=$
- Inférieur : $<$
- Supérieur : $>$
- Inférieur ou égal : $<=$
- Supérieur ou égal : $>=$



Les opérations logiques

- Et : &&
- Ou : ||
- Non : !



La structure if


```
if (condition1) {  
    // instructions à exécuter si la condition1 est vraie  
} else if (condition2) {  
    // instructions à exécuter si la condition2 est vraie  
} else {  
    // instructions à exécuter si aucune des conditions précédentes n'est vraie  
}
```



La structure switch

```
switch (expression) {  
    case valeur1:  
        // instructions à exécuter si expression est égale à valeur1  
        break;  
    case valeur2:  
        // instructions à exécuter si expression est égale à valeur2  
        break;  
    default:  
        // instructions à exécuter si aucune des valeurs précédentes n'est égale à expression  
}  

```



La boucle for

```
for (initialisation ; condition ; incrémentation) {  
    // instructions à répéter tant que la condition est vraie  
}
```

exemple :

```
for (int i = 0 ; i < 10 ; i++) {  
    System.out.println("Patient numéro" + i)  
}
```



La boucle while

```
while (condition) {  
    // instructions à répéter tant que la condition est vraie  
}
```





Scanner

La fonction Scanner

La fonction scanner permet de récupérer ce que l'utilisateur écrit dans la console.

Elle nécessite d'être importé : `import java.util.Scanner ;`

D'être définie : `Scanner scanner = new Scanner(System.in) ;`

Exemple : `String nom = scanner.nextLine() ;`



Les Tableaux

Déclaration d'un tableau

```
int[] tableauEntiers; // déclare un tableau  
d'entiers
```

```
String[] tableauChaines; // déclare un  
tableau de chaînes de caractères
```



Initialisation d'un tableau

Initialisation explicite

```
int[] tableauEntiers = {1, 2, 3, 4, 5};
```

```
String[] tableauChaines = {"rouge", "vert", "bleu"};
```

Initialisation dynamique

```
int[] tableauEntiers = new int[2];
```

```
tableauEntiers[0] = 1;
```

```
tableauEntiers[1] = 2;
```



Accéder aux éléments d'un tableau

```
int[] tableauEntiers = {1, 2, 3, 4, 5};
```

```
System.out.println(tableauEntiers[0]); // affiche 1
```

```
System.out.println(tableauEntiers[2]); // affiche 3
```



Tableaux multidimensionnels

Un tableau multidimensionnel est un tableau qui contient d'autres tableaux.
Par exemple, un tableau à deux dimensions peut être utilisé pour stocker des données sous forme de grille ou de matrice.

```
int[][] tableau2D = new int[5][5];
```



ArrayList

Permet de créer des tableaux de taille variable

```
ArrayList<String> liste = new ArrayList<String>()
```



Fonctions de l'ArrayList

`liste.size()` pour la taille

`liste.get(0)` pour récupérer la string à l'indice 0

`liste.add("test")` pour ajouter une string en fin de liste



Les fonctions

Définir une fonction

```
private static void sayHelloTo(String recipient) {  
    System.out.println("Hello " + recipient);  
}
```



Les objets

Définir un objet

```
public class Personne {
```

```
    private String nom;
```

```
    private String prenom;
```

```
    private String dateDeNaissance;
```

```
    public Personne(String nom, String prenom, String dateDeNaissance) {
```

```
        this.nom = nom;
```

```
        this.prenom = prenom;
```

```
        this.dateDeNaissance = dateDeNaissance;
```

```
    }
```



Définir des getters et setters

```
public String getNom() {
```

```
    return nom;
```

```
}
```

```
public void setNom(String nom) {
```

```
    this.nom = nom;
```

```
}
```

