

# Java

Les bases pour bien démarrer

## CONTENU

Fonctionnement de Java .....	1
Java Development Kit.....	1
Le langage JAVA .....	1
Les lignes de code .....	2
La compilation .....	2
L'exécution .....	2
Les fichiers .JAR.....	2
Introduction aux programmes JAVA .....	3
Les espaces de noms.....	3
Différencier les programmes .....	3
Regrouper les composants .....	4
Résumé du fonctionnement de JAVA.....	4
Démarrer avec JAVA et Visual Studio Code .....	5
Configurer mon espace de travail.....	5
Organiser mon espace de travail .....	6
Mon 1 <sup>er</sup> programme JAVA .....	7
Démarrer mon programme JAVA.....	8
Run.....	8
Debug.....	8
Conclusion.....	8

## FONCTIONNEMENT DE JAVA

Le langage JAVA permet de créer des applications multi-plateformes. Une application JAVA peut s'exécuter sur différents systèmes d'exploitation.

Les programmes écrits en JAVA sont exécutés dans un environnement d'exécution dédié appelé **JRE** pour **Java Runtime Environment**. Cet environnement doit être installé sur l'ordinateur pour exécuter de telles applications.

Le JRE est disponible sur le site officiel : <https://www.java.com/fr/download/manual.jsp>

## JAVA DEVELOPMENT KIT

Pour développer des programmes en JAVA, le **JDK** pour **Java Development Kit** est nécessaire.

Le JDK est disponible sur le site officiel : <https://www.oracle.com/fr/java/technologies/javase-downloads.html>

Lorsque vous installez le JDK, le JRE est inclus. Inutile donc de télécharger les deux.

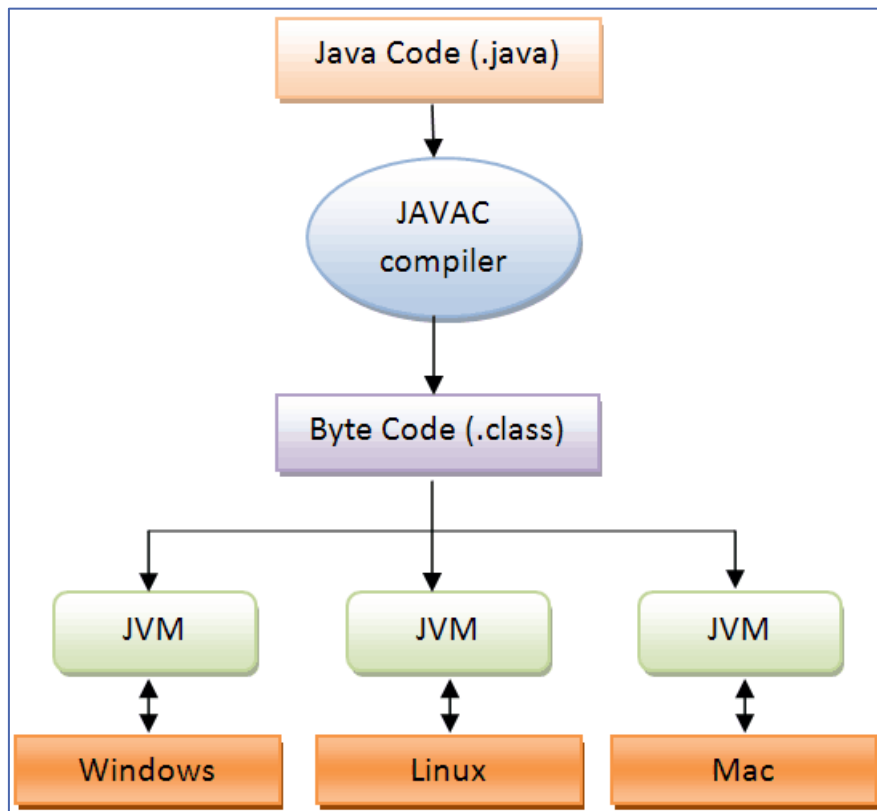
Exécuter des programmes JAVA	JRE – Java Runtime Environment
Exécuter des programmes JAVA + Développer des programmes JAVA	JDK – Java Development Kit

## LE LANGAGE JAVA

Le langage JAVA est utilisé pour la création de programmes JAVA.

Typiquement :

1. Les développeurs écrivent les lignes de code en Java.
2. Le code est compilé en "bytecode".
3. Le bytecode est exécuté dans la machine virtuelle de Java (JVM).



## LES LIGNES DE CODE

La 1<sup>ère</sup> étape consiste à écrire les lignes de code :

```
public class Hello
{
    Run | Debug
    public static void main(String[] args)
    {
        System.out.println("Hello world!");
    }
}
```

## LA COMPILATION

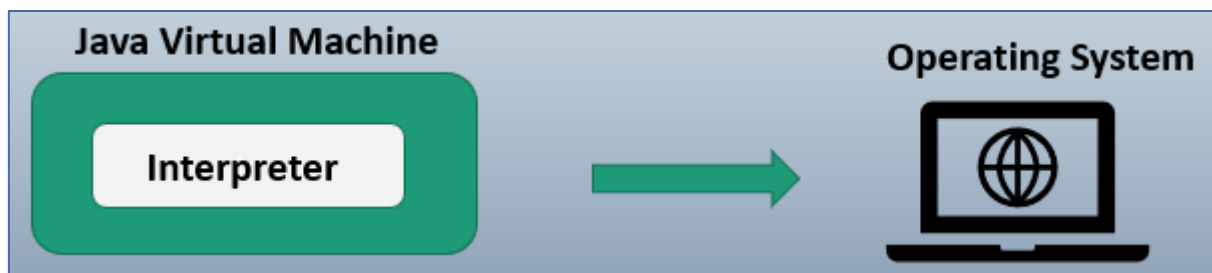
La **compilation** consiste à traduire le code écrit par les développeurs en un code plus bas niveau afin qu'il soit compris par la machine virtuelle de Java.

En effet, Java est un langage dit "compilé", il est donc nécessaire de passer par cette phase de compilation avant de pouvoir utiliser le programme.



## L'EXECUTION

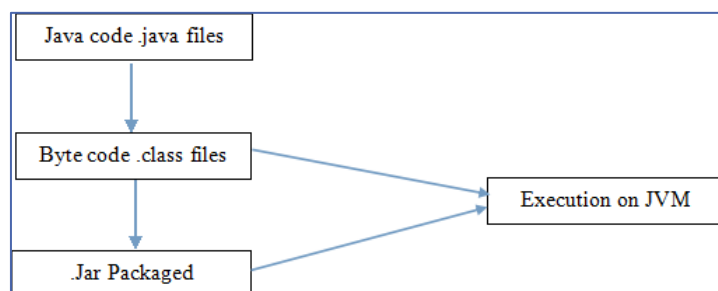
Les programmes JAVA sont **compilés** en un code intermédiaire appelé "**bytecode**". Lorsque le programme est exécuté sur un ordinateur, ce bytecode est **interprété** par la machine virtuelle de JAVA qui se chargera de le compiler en code machine (binaire) et d'exécuter le programme sur le système d'exploitation cible.



## LES FICHIERS .JAR

Un fichier JAR (Java ARchive) est un fichier ZIP utilisé pour distribuer un ensemble de classes Java. Ce format est utilisé pour stocker les définitions des classes, ainsi que des métadonnées, constituant l'ensemble d'un programme JAVA.

Un fichier JAR est directement exécutable sur la machine hôte.



## INTRODUCTION AUX PROGRAMMES JAVA

Un programme JAVA est constitué d'un ou plusieurs fichiers source contenant le code du programme.

Un fichier **.java** est un fichier de code JAVA. Chaque fichier contient un composant JAVA nommé **classe** (1 fichier = 1 composant).

Un programme JAVA contient toujours un composant spécial correspondant au "point d'entrée" du programme ; c'est-à-dire le code qui sera exécuté en premier au lancement du programme.

Ce composant doit contenir une méthode de classe appelé **main** et acceptant un argument de type **String[]**.

Reprenons l'exemple de code précédent :

```
public class Hello
{
    Run | Debug
    public static void main(String[] args)
    {
        System.out.println("Hello world!");
    }
}
```

Ce code correspond à la classe Hello qui contient le point d'entrée du programme.

Le code à l'intérieur la méthode **main** sera exécuté au lancement du programme et affichera ici "Hello world!" dans la console.

## LES ESPACES DE NOMS

Il existe une multitudes de programmes écrits en JAVA. Pour les différencier et également pour regrouper les composants d'une même famille, le code est structuré en "espaces de noms".

Un espace de noms correspond plus ou moins au chemin vers le composant et donc généralement à un répertoire sur l'ordinateur. Par exemple, l'espace de noms "helloworld" correspond à un dossier "helloworld" dans le répertoire de travail de mon application JAVA.

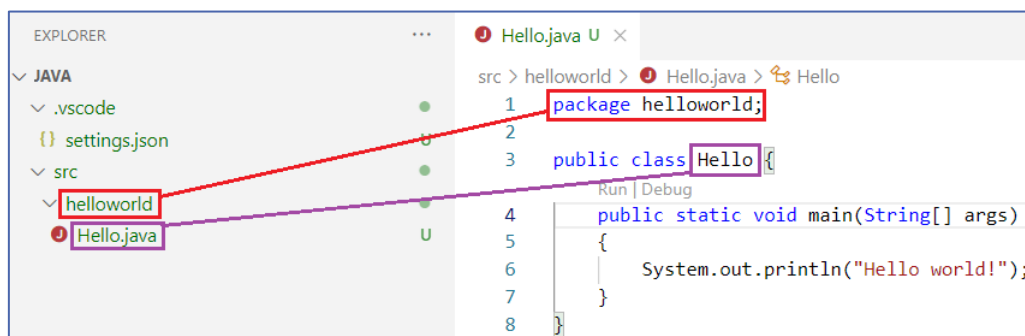
Tous les composants liés à un espace de noms doivent être placés à l'intérieur du répertoire correspondant.

## DIFFERENCIER LES PROGRAMMES

Pour bien différencier les programmes chacun d'entre eux possède son propre espace de noms. Par exemple, "calculatrice", "gestionemployes" ou "tetris". En Java, le mot clé "package" est utilisé pour définir l'espace de noms d'un fichier Java.

Exemple :

Considérons le répertoire "src" comme mon répertoire de travail Java.



Le fichier Hello.java se trouve dans le répertoire "helloworld"

→ déclaration du package "helloworld"

Le fichier Hello.java contient le composant "Hello"

→ déclaration de la classe "Hello"

## REGROUPER LES COMPOSANTS

Un programme est généralement constitué de plusieurs classes. Certaines d'entre elles, similaires, peuvent être regroupées en un espace de noms devenant un espace de noms enfant du précédent.

Dans la capture ci-dessous, remarquez la correspondance entre :

- La structure des répertoires et les espaces de noms (package) déclarés dans les fichiers
- Le nom des fichiers et le nom des classes

```

EXPLORER
  JAVA
    .vscode
    {} settings.json
    src
      > helloworld
      > people
        > models
          Person.java
          App.java
          README.md

src > people > App.java
1 package people;
2
3 import people.models.Person;
4
5 public class App {
6
7     public static void main(String[] args) {
8         Person p1 = new Person("Mike");
9         Person p2 = new Person("Guy");
10
11         System.out.println(p1.getName());
12         System.out.println(p2.getName());
13     }
14 }
15
16

src > people > models > Person.java
1 package people.models;
2
3 public class Person {
4
5     private String name;
6
7     public Person(String _name)
8     {
9         this.name = _name;
10    }
11
12    public String getName()
13    {
14        return this.name;
15    }
16 }
17
  
```

Vous remarquez également la présence d'une instruction "import" qui permet à des classes d'utiliser d'autres classes se situant dans d'autres espaces de noms, c'est la notion de dépendance.

Dans la capture précédente,

- Ligne 3 : l'instruction import indique dans quel espace de noms se situe la classe "Person"
- Lignes 8 et 9 : utilisation de la classe "Person".

Les notions de dépendances sont abordées dans le module "Modélisation Objet".

## RESUME DU FONCTIONNEMENT DE JAVA

- Le Java Runtime Environment (JRE) permet d'exécuter des programmes JAVA.
  - Le Java Development Kit (JDK) permet de développer des programmes avec le langage JAVA.
  - Le JDK contient le JRE.
  - Le code JAVA doit être compilé en bytecode avant de pouvoir être exécuté.
  - La machine virtuelle de JAVA (JVM) interprète le bytecode et exécute le programme sur le système d'exploitation.
  - Les programmes complexes sont compilés dans des fichiers Java Archive (.jar).
- 
- Le code JAVA est structuré en espaces de noms (package).
  - La structure des fichiers source d'un programme JAVA suit la logique des espaces de noms :
    - Répertoire = espace de noms (package)
    - Fichier = Composant (classe)
  - Les espaces de noms s'écrivent en minuscules
  - Les noms de classes s'écrivent en StudyCase (ou PascalCase)
    - 1<sup>ère</sup> lettre en majuscule, les lettres suivantes en minuscules
      - Exemples: Hello, App, Voiture
    - Pour les noms composés, la 1<sup>ère</sup> lettre de chaque mot est en majuscule
      - Exemples: MaClasse, VoitureDeCourse, HelloWorld.

## DEMARRER AVEC JAVA ET VISUAL STUDIO CODE

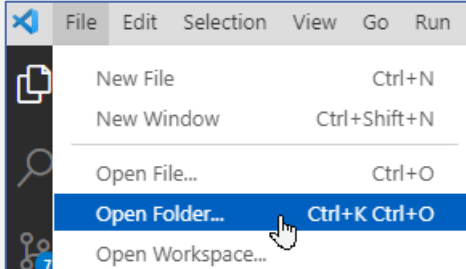
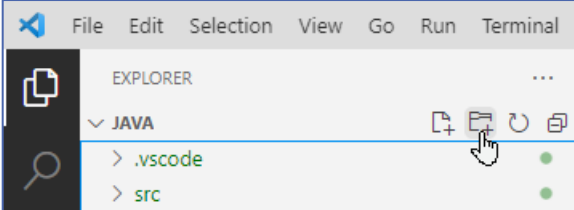
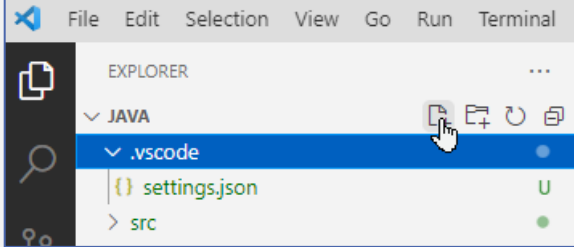
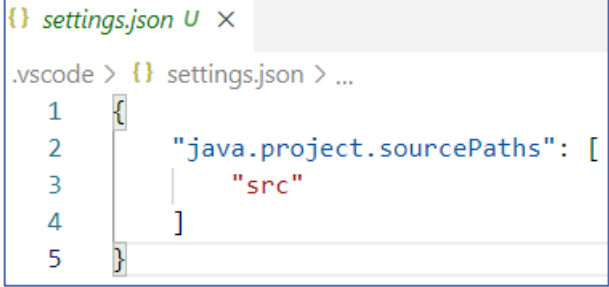
Visual Studio Code est un éditeur de texte avancé proposant de nombreuses fonctionnalités utiles aux développeurs.

Pour développer en Java avec Visual Studio Code, vous devez installer les logiciels nécessaires que vous trouverez facilement sur le web. Tous les outils proposés ci-dessous sont gratuits.

Installez, dans cet ordre :

1. Java Development Kit (JDK ou openJDK)
  - <https://www.oracle.com/fr/java/technologies/javase-downloads.html>
2. Visual Studio Code
  - <https://code.visualstudio.com>
  - Si vous ne possédez pas les droits administrateur sur votre machine, téléchargez le "user installer"
3. Java Extension Pack pour Visual Studio Code
  - <https://marketplace.visualstudio.com/items?itemName=vscjava.vscode-java-pack>

## CONFIGURER MON ESPACE DE TRAVAIL

<p>Créez un répertoire sur votre PC et ouvrez le dans Visual Studio Code. Ce répertoire est votre <b>espace de travail</b>.</p> <p>(Fichier -&gt; Ouvrir un dossier)</p> <p>Dans les captures suivantes, ce dossier s'appelle "java".</p>	
<p>Dans ce dossier, créez 2 répertoires :</p> <ul style="list-style-type: none"> <li>• .vscode</li> <li>• src</li> </ul>	
<p>Dans le dossier ".vscode" créez le fichier settings.json.</p> <p>Consulter le PDF "Le format JSON" pour en apprendre plus sur JSON.</p>	
<p>Ouvrez le fichier settings.json et ajoutez le code correspondant <b>exactement</b> à la capture ci-contre.</p> <p>Le code ci-contre indique à Visual Studio Code que le dossier "src" se situant dans l'<b>espace de travail</b> est votre <b>répertoire de travail JAVA</b>. Cela vous sera utile pour exécuter et debugger votre code directement depuis Visual Studio Code.</p>	 <pre> 1 { 2   "java.project.sourcePaths": [ 3     "src" 4   ] 5 }</pre>

Visual Studio Code est désormais prêt pour votre démarrage avec JAVA.

## ORGANISER MON ESPACE DE TRAVAIL

Le répertoire "src" contiendra vos différents programmes.

Vous pouvez créer un espace de travail complet (en suivant la procédure précédente) par programme.

ou

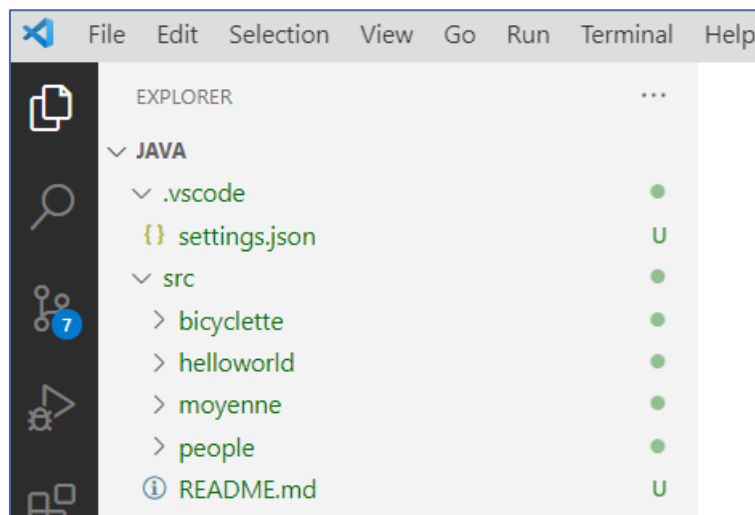
Vous pouvez créer un espace de travail et y regrouper plusieurs programmes.

Dans le cadre de la formation, nous vous recommandons d'utiliser la seconde solution afin de regrouper vos exercices dans un espace de travail commun.

- un espace de travail pour les exercices "Algorithmes"
- un espace de travail pour les exercices "Objets"

Exemple :

La capture suivante montre plusieurs programmes dans le répertoire de travail JAVA :

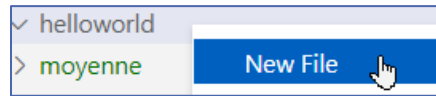


Un dossier correspond à un exercice/programme.

Chacun des dossiers contient une classe de démarrage.

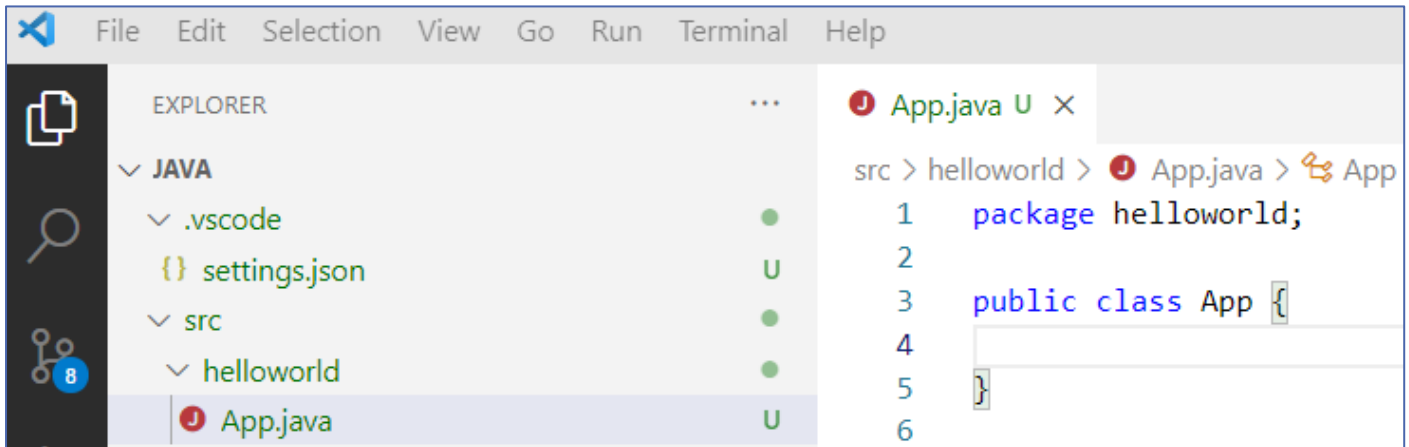
## MON 1<sup>ER</sup> PROGRAMME JAVA

Dans le répertoire "src" de votre espace de travail, créez un dossier portant le nom de votre programme/exercice, par exemple "helloworld". Puis cliquez droit sur ce dossier et sélectionnez "nouveau fichier".



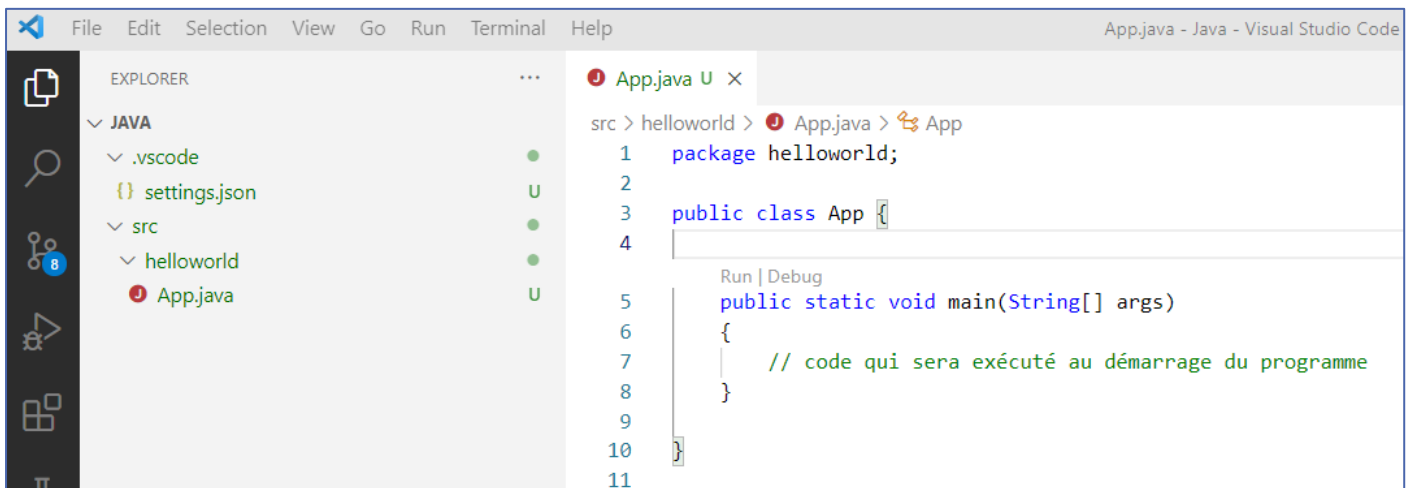
Nommez ce fichier App.java (par convention, le point d'entrée est le composant "App").

Si vous avez bien installé les extensions JAVA, Visual Studio Code générera le squelette du code de votre composant avec le nom du package correspondant au répertoire et le nom de la classe correspondant au fichier :

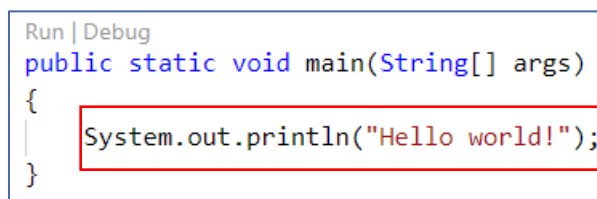


Si tel n'est pas le cas, vérifiez l'installation du Java extension Pack ou ajoutez le code de la capture précédente dans votre fichier App.java.

Ce fichier sera le point d'entrée du programme, vous devez donc y ajouter la méthode de classe main :



A l'intérieur de la méthode "main", ajoutez la ligne suivante (encadrée en rouge) :

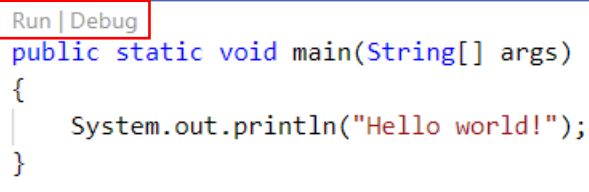


Cette ligne affichera "Hello world" dans le terminal qui exécutera le programme.

**Note :** Un programme JAVA ne peut contenir qu'un seul composant avec une méthode de classe main.

## DEMARRER MON PROGRAMME JAVA

Si vous avez bien suivi les étapes d'installation de Visual Studio Code et de Java, vous devriez remarquer la présence de 2 petits liens "Run" et "Debug" juste au-dessus de votre méthode main.



```
Run | Debug
public static void main(String[] args)
{
    System.out.println("Hello world!");
}
```

## RUN

Cliquer sur "Run" exécutera votre programme dans le terminal intégré à Visual Studio Code.

Visual Studio Code se chargera de compiler votre programme et de l'exécuter dans la JVM.

## DEBUG

Cliquer sur "Debug" exécutera votre programme en mode **débogage** dans le terminal intégré à Visual Studio Code.

Le débogage vous permettra de suivre l'exécution de votre programme et d'effectuer des opérations de contrôles sur votre code et ce, pendant l'exécution du programme.

## CONCLUSION

Vous disposez maintenant des bases pour démarrer le développement JAVA avec Visual Studio Code.

De nombreux autres éditeurs de code et IDE (Environnement de Développement intégré) permettent de travailler avec Java.

Cette introduction avec VScode vous aura permis de comprendre la structure d'un programme JAVA.

Vous pouvez désormais commencer votre initiation avec JAVA 😊.

### Pour démarrer avec le langage JAVA :

- Consulter le fichier PDF "Java 02"
- Suivez ce tutoriel : <https://www.ukonline.be/cours/java/apprendre-java>

--- FIN DU DOCUMENT ---