



Chapitre 1

Les langages de programmation

Vous venez de découvrir ce que sont les algorithmes. Comment nous l'avons mentionné précédemment, un algorithme se doit d'être indépendant d'un langage de programmation. Il est maintenant temps pour vous de commencer à traduire les algorithmes que vous avez conçu en utilisant un langage de programmation.

Dans ce chapitre, nous allons débiter en définissant ce qu'est un langage de programmation. Par la suite, nous mettrons en place nos outils de travail puis, nous découvrirons comment se compose une première application, le fameux Hello world, utilisé comme application lors d'une première utilisation d'un langage informatique.

OBJECTIFS

- Définir c'est quoi un langage de programmation.
- Créer une première application informatique.

[illegible]



Leçon 1-A

Qu'est-ce qu'un langage de programmation ?

Un langage de programmation est une notation conventionnelle destinée à formuler des algorithmes et produire des programmes informatiques qui les appliquent. D'une manière similaire à une langue naturelle, un langage de programmation est composé d'un alphabet, d'un vocabulaire, de règles de grammaire, de significations, mais aussi d'un environnement de traduction censé rendre sa syntaxe compréhensible par la machine.

Source : https://fr.wikipedia.org/wiki/Langage_de_programmation

OBJECTIFS

À la fin de la présente leçon, vous serez en mesure :

- de définir c'est quoi un langage de programmation.

Vous ne les voyez presque jamais, et pourtant ils vous permettent d'utiliser logiciels et applications. Sans eux, pas de Facebook, pas d'AirBnB, ou d'Uber Eats. Sans eux, pas de filtres Insta ou d'intelligence artificielle capable de dire si sur votre photo se trouve un chat ou un écureuil. Vous avez sûrement déjà entendu parler d'HTML, et la tasse de café du logo de Java vous évoque peut-être quelque chose. Pourtant, les langages de programmation sont à la base de l'écosystème numérique.

- *Alors, de quoi parle-t-on concrètement ?*
- *Qu'est-ce qu'un langage de programmation ?*

Débutons avec une définition, celle d'un programme informatique. Un **programme informatique** est une séquence d'instructions textuelles destinées à être exécutées par un ordinateur. Autrement dit, la **programmation informatique** est le fait de matérialiser une suite d'algorithmes pour qu'ils soient convertis en langage machine puis exécutés par un ordinateur.

Chaque programme est écrit par un développeur dans un **langage de programmation**. Un langage de programmation, c'est donc simplement une notation conventionnelle destinée à traduire des algorithmes en vue de produire les programmes informatiques qui les appliquent. Les langages de programmation fonctionnent comme n'importe quelle langue naturelle, ils permettent d'exprimer une abstraction grâce à un alphabet, un vocabulaire, des règles de grammaire et de significations.

Concrètement, les langages de programmation permettent :

- De décrire les structures des données qui seront manipulées par l'appareil informatique, c'est-à-dire les façons selon lesquelles les données seront organisées et stockées afin d'être traitées.
- D'indiquer comment sont effectuées les manipulations, c'est-à-dire selon quels algorithmes.

Les composantes d'un langage de programmation

Règles de syntaxe	Combinaison des unités linguistiques afin qu'elles aient un sens (ponctuation, par exemple)
Lexique	Instructions construites à partir de symboles : <ul style="list-style-type: none">– les mots, à partir de lettres : if, let, then, else...– les opérateurs mathématiques : +, -, * ...– les booléens
Sémantique	Sens de chacune des phrases qui peut être construite dans un langage (que va-t-il se passer si j'écris ceci ?)
Alphabet	Par exemple, l'alphabet classique American Standard Code for Information Interchange, code ASCII (chiffres arabes de 0 à 9, lettres minuscules et majuscules de A à Z, symboles mathématiques et de ponctuation) ou l'alphabet Unicode.
Commentaires	Textes non traduits et non exécutables qui permettent de laisser des explications.
Identifiants	Éléments constitutifs du programme qui permettent de le structurer (variables, procédures, types...).

Nous avons déjà couvert les notions de *variables* et de *constantes* ainsi que les types de données précédemment dans ce cours. Une *procédure*, ou *fonction* ou *méthode*, est un fragment de programme que l'on paramètre une fois et que nous sommes en mesure de réutiliser si besoin. Nous couvrirons plus en détail l'utilisation des *procédures* et des *fonctions* dans un prochain chapitre.

Langages de programmation et paradigmes

En logique, un *paradigme* est un modèle théorique de pensée qui oriente la recherche et la réflexion scientifiques. Un *paradigme de programmation* est la vision avec laquelle le programmeur aborde la programmation selon des concepts, des théories. C'est un peu le style du programme. Plus concrètement, c'est la manière dont les solutions aux problèmes doivent être formulées dans tel ou tel langage de programmation. Il n'existe pas une théorie permettant de couvrir tous les concepts de la programmation, ce qui implique une variété de paradigmes. Toutefois, la majorité des langages de programmation sont aujourd'hui multiparadigmes. Parmi les plus utilisés, il y a :

Le *paradigme impératif*, comparable à une recette de cuisine, on exécute les actions successives, lignes de code, qui nous permettent de modifier l'état des ingrédients, du programme, jusqu'à aboutir à leur état final, leur plat, la sortie du programme.

Exemples : C, C++, Pascal, Java, PHP, JavaScript, Python...

Le *paradigme déclaratif*, énonce le résultat final souhaité sans décrire l'ordre des instructions intermédiaires pour y parvenir. Les instructions intermédiaires sont déterminées automatiquement.

Exemples : HTML (déclaratif descriptif), Haskell (déclaratif fonctionnel), Prolog (déclaratif logique) ...

Le *paradigme événementiel*, le programme est exécuté lorsqu'un événement se passe par exemple, la *souris est déplacée*, le *bouton gauche de la souris est pressé*. Il a été introduit par le langage Simula dans les années 1970.

Le *paradigme visuel*, permet de manipuler des objets bidimensionnels, par exemple des cases, des flèches, par glisser-déposer. Le dessin obtenu est ensuite traduit en programme orienté objet et événementiel.

Exemples : Smalltalk, Scratch, Snap

Le *paradigme orienté objet*, vise à définir et faire interagir entre eux des objets, compris ici comme tous types de structures issues d'un langage donné. L'analogie avec le monde réel est simple. Prenez l'objet vélo. Ses propriétés sont les suivantes : deux roues, des pédales, couleur rouge. L'objet vélo peut réaliser l'action rouler et il peut interagir avec l'objet cycliste, qui le fait rouler, et l'objet guidon, qui lui permet de tourner.

Exemples : Java, C#, Python, Ruby, JavaScript, C++, PHP

La *programmation orientée objet* a joué un grand rôle dans la démocratisation de la science des données, en particulier grâce à la création de bibliothèques, qui permettent d'obtenir des modules, i.e. des *bouts de code*, et de les utiliser dans son programme sans avoir besoin de les coder, et de la manière que l'on souhaite en fonction de nos besoins. La *programmation orientée objet* sera traitée plus en détail plus tard dans le cadre de votre formation.

Il existe des centaines et des centaines de langages de programmation. Pour vous donner une idée, visitez le site suivant afin d'en avoir un aperçu : https://fr.wikipedia.org/wiki/Liste_de_langages_de_programmation.



Se remémorer tous les noms qui figurent sur ce site tient du miracle et, prétendre vouloir maîtriser cinq ou six de ces langages dans le temps que vous disposez pour votre formation est impensable et inconcevable.

Comme dans toute bonne formation digne de ce nom, il faut débiter par la base, ce que nous avons commencé à voir dans les chapitres précédents en étudiant ce qu'est l'*algorithmique*. Vous devez maîtriser la création d'algorithme avant même de penser à utiliser un langage de programmation afin de créer une application informatique.

Développer un programme informatique ne signifie pas d'être en mesure d'écrire, en utilisant un langage de programmation, les instructions qui seront exécutées par l'ordinateur afin d'obtenir le résultat souhaité. **Développer un programme informatique** c'est avant tout une façon de penser, ce pourquoi les notions d'algorithmique vous ont été présentées précédemment. Vous devez prendre la bonne habitude de préparer votre algorithme et de le tester manuellement avant de penser à écrire les instructions que vous désirez que l'ordinateur effectue.

Si vous arrivez à maîtriser la création d'algorithmes fonctionnels, le langage de programmation que vous utiliserez pour créer votre application ne sera que plus facile à apprendre en utilisant différentes sources d'information comme des livres spécialisés sur un langage précis, des vidéos sur un site comme YouTube ou des sites de formations spécialisées comme Udemy ou OpenClassroom. Dans votre situation, en tant qu'étudiant, la personne qui devrait compter le plus pour vous est, vous-même, supporté par votre instructeur.

Récapitulation

Un ordinateur ne comprend qu'un seul langage, le langage binaire, une suite de 0 et de 1 qu'un programmeur n'est pas en mesure de comprendre.

Un langage de programmation est une notation conventionnelle destinée à formuler des algorithmes et produire des programmes informatiques qui les appliquent.

Un langage de programmation est composé des éléments suivants :

Règles de syntaxe	Combinaison des unités linguistiques afin qu'elles aient un sens (ponctuation, par exemple)
Lexique	Instructions construites à partir de symboles : – les mots, à partir de lettres : if, let, then, else... – les opérateurs mathématiques : +, -, * ... – les booléens
Sémantique	Sens de chacune des phrases qui peut être construite dans un langage (que va-t-il se passer si j'écris ceci ?)
Alphabet	Par exemple, l'alphabet classique American Standard Code for Information Interchange, code ASCII (chiffres arabes de 0 à 9, lettres minuscules et majuscules de A à Z, symboles mathématiques et de ponctuation) ou l'alphabet Unicode.
Commentaires	Textes non traduit et non exécutable qui permettent de laisser des explications.
Identifiants	Éléments constitutifs du programme qui permettent de le structurer (variables, procédures, types...).

Notes

[illegible]



Leçon 1-B

Préparation de notre environnement de développement

Maintenant que nous savons c'est quoi un langage de programmation, il serait temps pour nous de commencer à en utiliser un. Mais, avant de s'y attaquer, nous allons débiter avec la préparation de notre environnement de travail afin de pouvoir utiliser correctement le langage de programmation que nous allons utiliser dans ce cours.

OBJECTIFS

À la fin de la présente leçon, vous serez en mesure :

- de préparer votre environnement de travail.

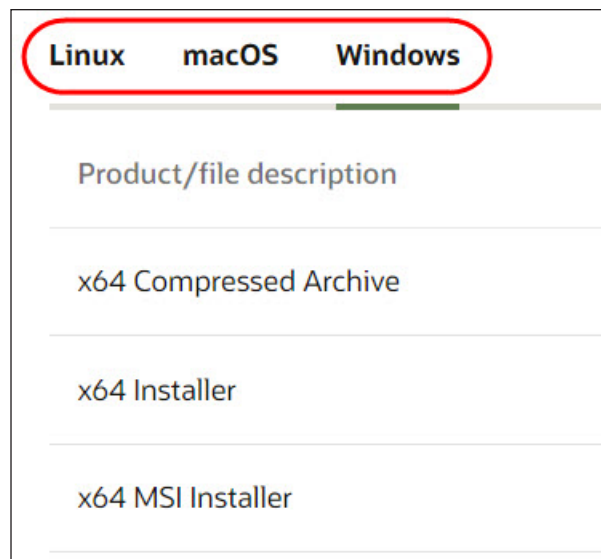
Avant de commencer à utiliser un langage de programmation, il est important que nous débutions avec la configuration de notre environnement de développement sur notre ordinateur. La première chose à faire est de choisir l'IDE, *Integrated Development Environment* ou environnement de développement intégré, que vous utiliserez pour développer avec le langage de programmation choisi. Dans ce cours et les suivants, nous utiliserons le langage de programmation Java.

Un IDE est une application dans laquelle vous pouvez saisir, compiler et exécuter votre code source. L'IDE essaiera également de vous aider dans votre travail en fournissant un IntelliSense, qui offre des outils tels que la complétion de code, des informations sur les paramètres, des listes de suggestions et bien plus encore. Visual Studio Code est un bon exemple d'IDE.

Il existe de nombreux IDE pour coder en Java. Ce cours s'efforcera d'être agnostique en ce qui concerne les IDE, ce qui signifie que pratiquement tout devrait fonctionner quel que soit l'IDE que vous choisirez. Cependant, comme vous êtes déjà familier avec Visual Studio Code, c'est l'IDE que nous utiliserons pour ce cours.

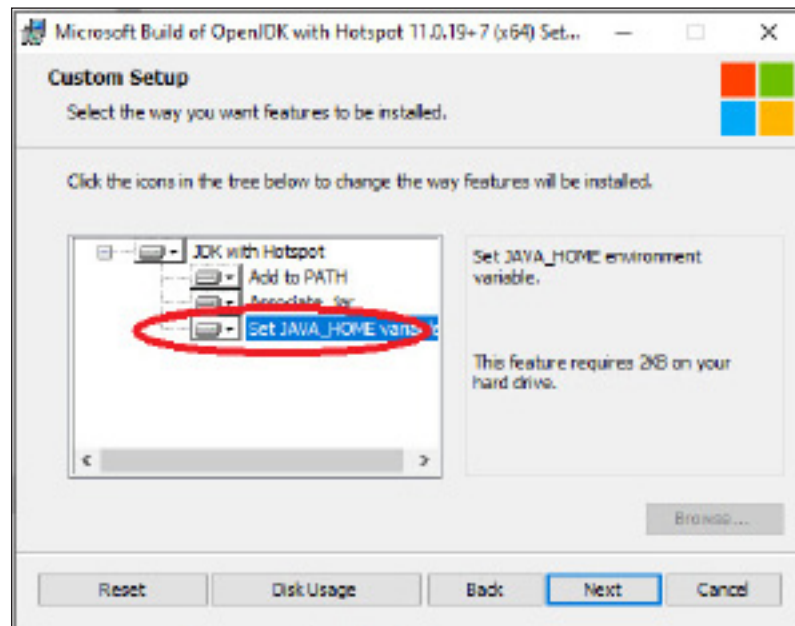
La préparation de notre ordinateur afin de pouvoir programmer en utilisant le langage Java se fait en deux étapes faciles. La première étape consiste à télécharger et à installer le kit de développement Java, JDK pour Java Development Kit. Le JDK contient de nombreux outils essentiels au développement en Java. Les outils les plus importants sont *javac.exe* et *java.exe*, qui sont les fichiers de commande permettant de compiler et d'exécuter vos applications écrites en Java. Nous reviendrons sur l'utilisation de ces outils dans le dernier chapitre de ce cours.

Il existe plusieurs sites vous permettant de télécharger le JDK. Parmi ceux-ci, le site officiel de l'entreprise Oracle, propriétaire du JDK à l'adresse suivante : <https://www.oracle.com/java/technologies/downloads/>. À partir de cette fenêtre, vous pouvez sélectionner le système d'exploitation de votre ordinateur et télécharger le JDK adapté à votre système. Lors de l'écriture de ces pages, c'était la version 20.0.2 du JDK qui était offert en téléchargement.



Vous pouvez télécharger la version du JDK que vous désirez. Dans mon cas, mon ordinateur étant équipé de Windows 11, j'ai choisi la version x64 MSI Installer. Lorsque le téléchargement est complété, vous pouvez procéder à l'installation du JDK sur votre ordinateur.

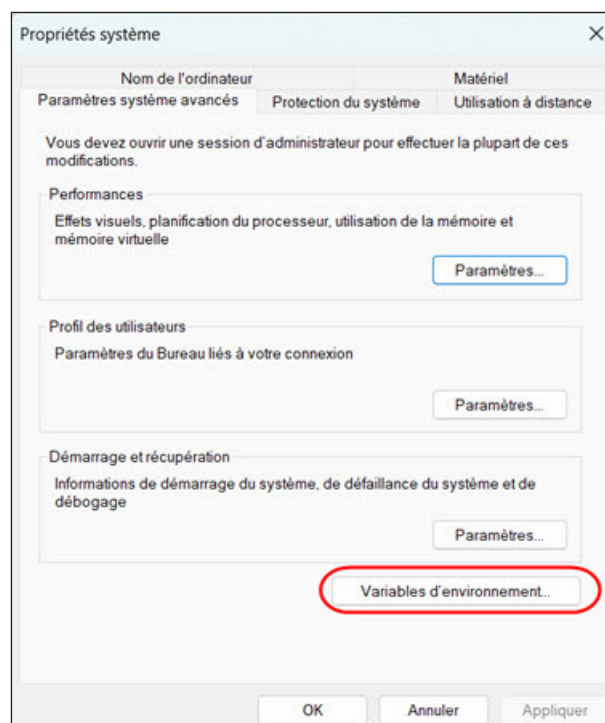
Lors de l'installation du JDK, il est recommandé de définir la variable d'environnement JAVA_HOME afin que le JDK soit facilement trouvé par votre IDE.



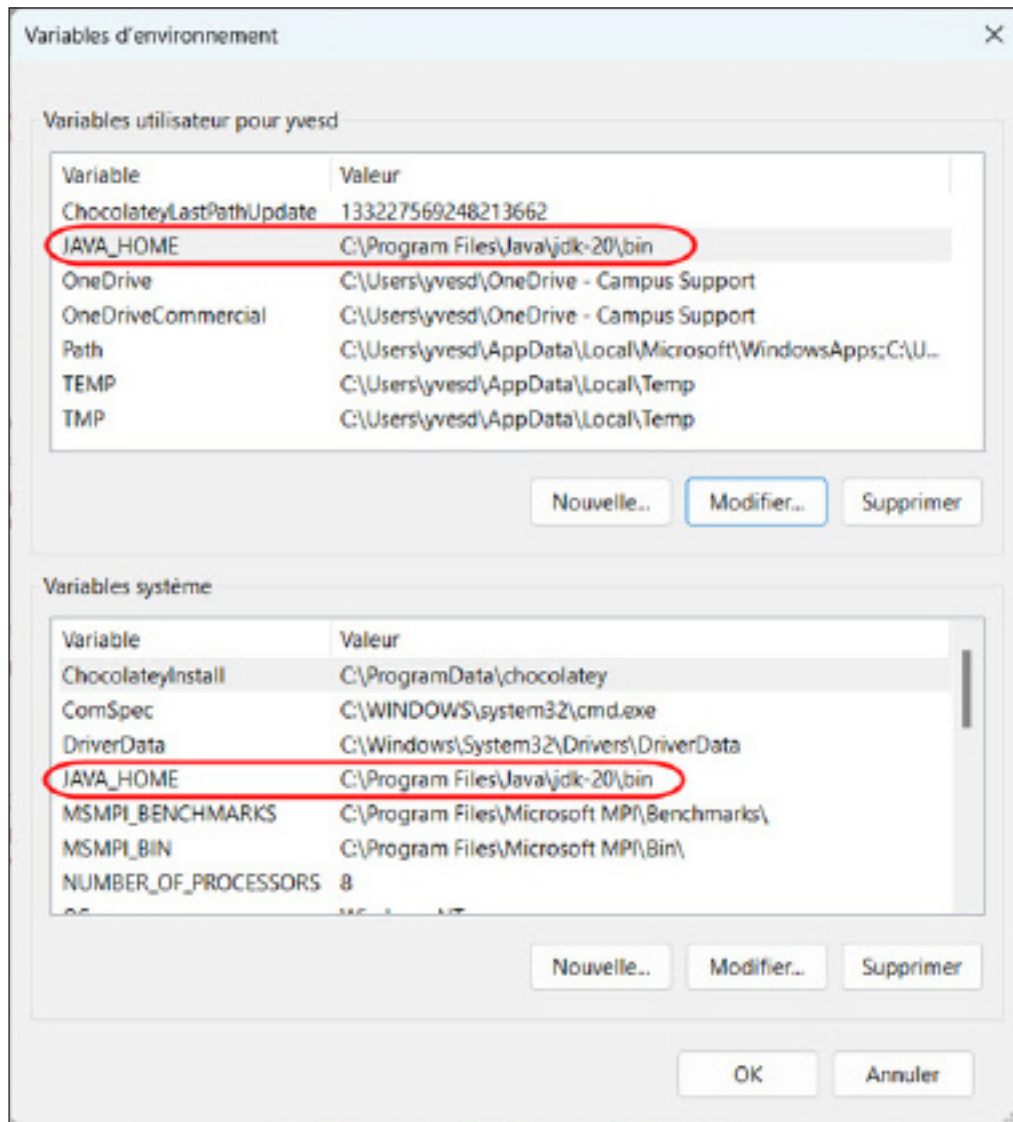
Il est possible que lors de l'installation l'option ne vous soit pas proposée, si tel est le cas, voici ce que vous devez faire.

Trouvez l'endroit où le dossier bin du JDK a été installé sur votre ordinateur, normalement sur la partition principale de votre ordinateur, dans le dossier Program Files, Fichiers de programme, sous-dossier Java\JDK. Dans mon cas, le chemin d'accès est le suivant : C:\Program Files\Java\jdk-20\bin.

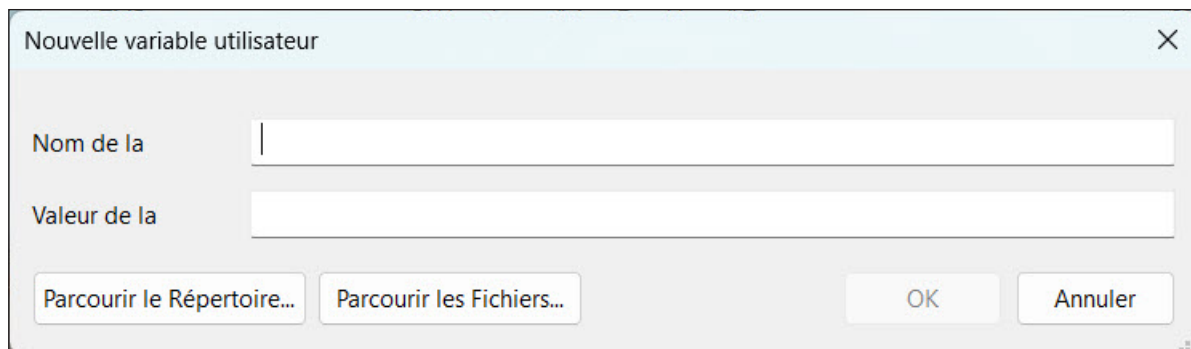
Copiez ce chemin d'accès puis, accédez à la fenêtre de configuration de vos variables d'environnement en utilisant la fenêtre *Propriétés du système* et en cliquant sur le bouton *Variables d'environnement*.



La fenêtre suivante s'affichera à l'écran.

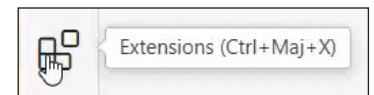


Dans cette dernière fenêtre, vérifiez si la variable JAVA_HOME existe. Si oui, assurez-vous que la valeur associée à votre variable concorde avec le chemin d'accès vers le fichier bin de l'installation que vous venez d'effectuer. Vous avez toujours la possibilité de modifier cette valeur en cliquant sur le bouton **Modifier...** sous chacune des deux fenêtres. Dans le cas où votre variable d'environnement n'existe pas, vous pouvez la créer en utilisant le bouton **Nouvelle...** La fenêtre suivante s'affichera à l'écran.



Dans cette fenêtre, utilisez les zones de saisie *Nom de la* et *Valeur de la* afin de saisir les informations puis cliquez sur le bouton **OK** pour enregistrer votre variable.

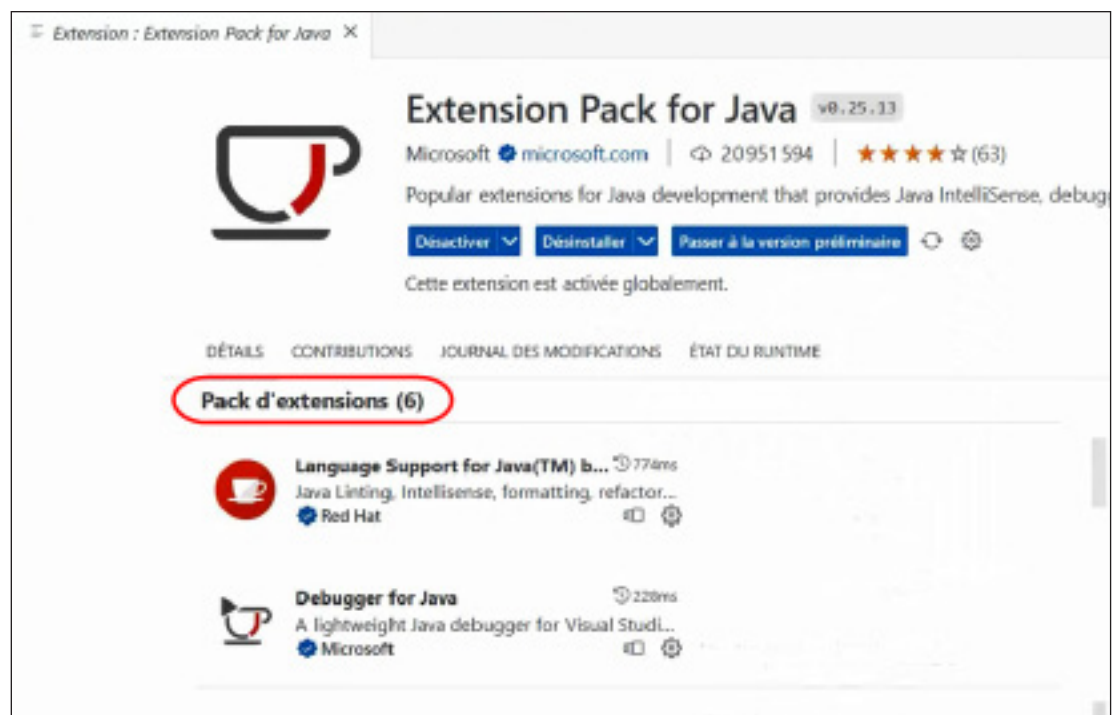
La dernière étape est de configurer Visual Studio Code afin que ce dernier soit en mesure de pouvoir être utilisé pour l'écriture du code Java. Démarrez Visual Studio Code et dans la marge gauche de l'IDE, cliquez sur l'icône Extensions.



Dans la zone de saisie de la section **EXTENSIONS** qui s'est affiché saisissez **java**. Une liste d'extensions pour Java s'affichera sous votre zone de saisie. Recherchez **Extension Pack for Java** dans cette liste et cliquez sur celui-ci.



Dans la nouvelle fenêtre qui s'affiche à droite, cliquez sur le bouton **Installer**. Dans l'exemple de la prochaine figure, ce bouton n'est pas présent étant donné que ce Pack d'extension est déjà installé sur mon ordinateur.



Vous remarquerez que ce Pack d'extensions installe six extensions différentes pour le langage Java. Cette seule installation vous permettra de pouvoir travailler aisément avec Visual Studio Code pour un développement en utilisant le langage Java. Notre ordinateur est maintenant prêt à être utilisé pour la suite de ce cours et des suivants.

Notes

[illegible]



Leçon 1-C

Nos premiers pas avec un langage de programmation

Dans la dernière leçon, nous avons pris le temps de préparer notre ordinateur afin d'être en mesure de pouvoir commencer à travailler le développement de nos programmes en utilisant le langage de programmation Java. Dans cette leçon nous allons découvrir notre environnement de développement avec la création d'une première application, le fameux Hello world utiliser dans toutes les formations de programmation.

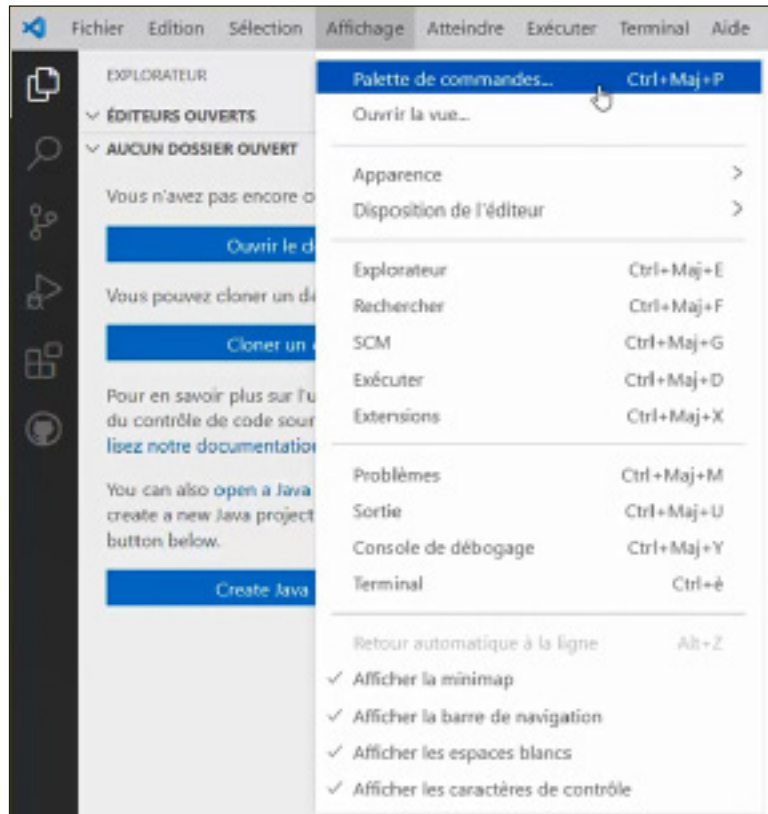
OBJECTIFS

À la fin de la présente leçon, vous serez en mesure :

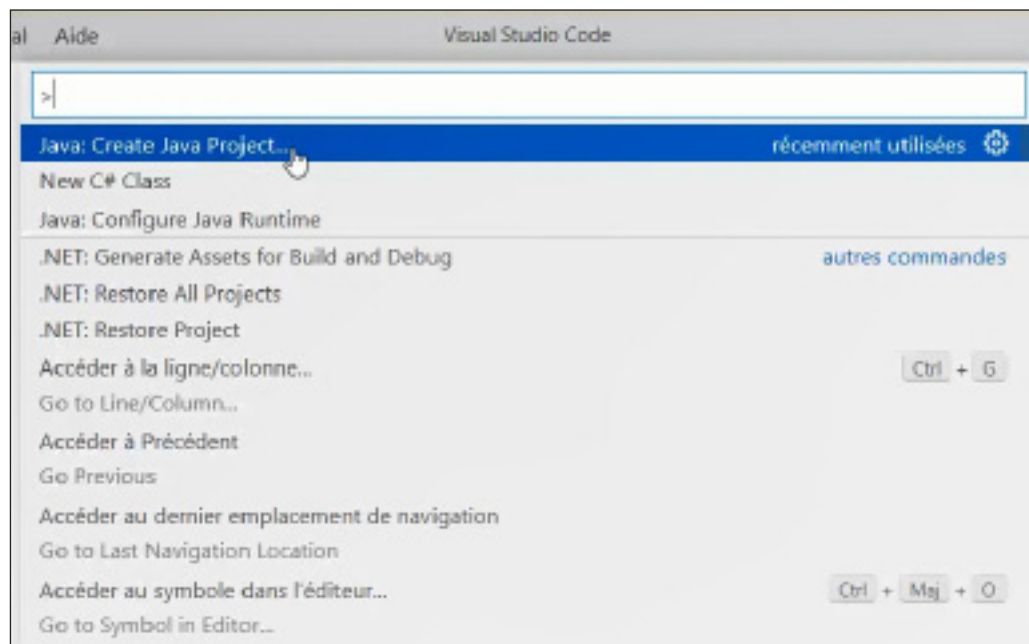
- de créer une première application en utilisant le langage de programmation Java.

Avant de commencer à programmer notre première application avec Java, nous devons créer notre projet. La création d'un projet en Java se fait en quelques étapes simples avec Visual Studio Code.

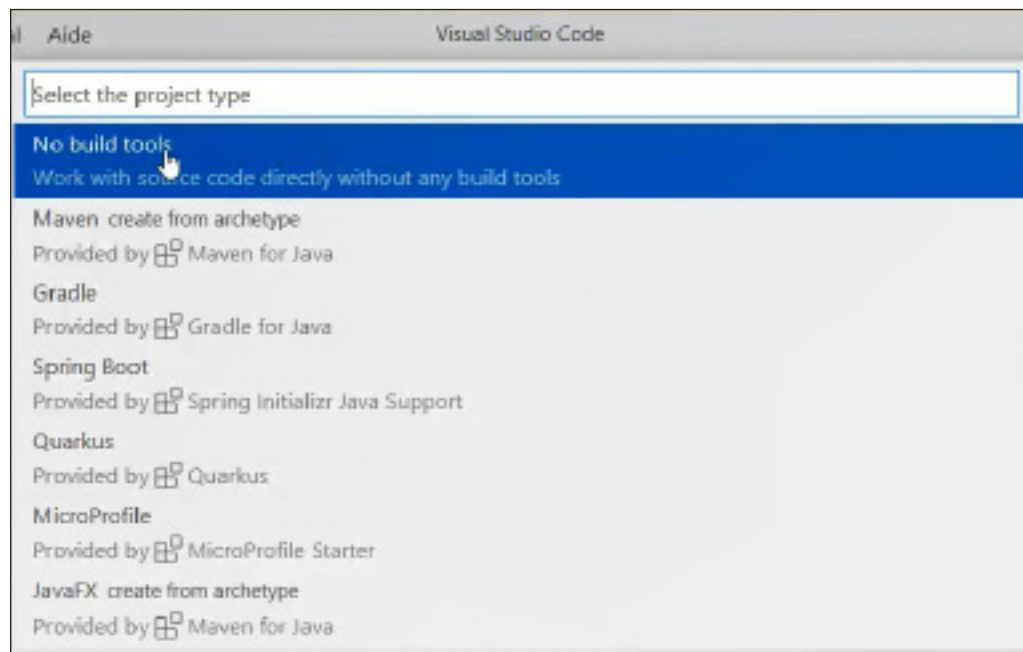
Cliquez sur l'élément de menu **Affichage** et sélectionnez l'élément **Palette de commandes** du sous-menu affiché.



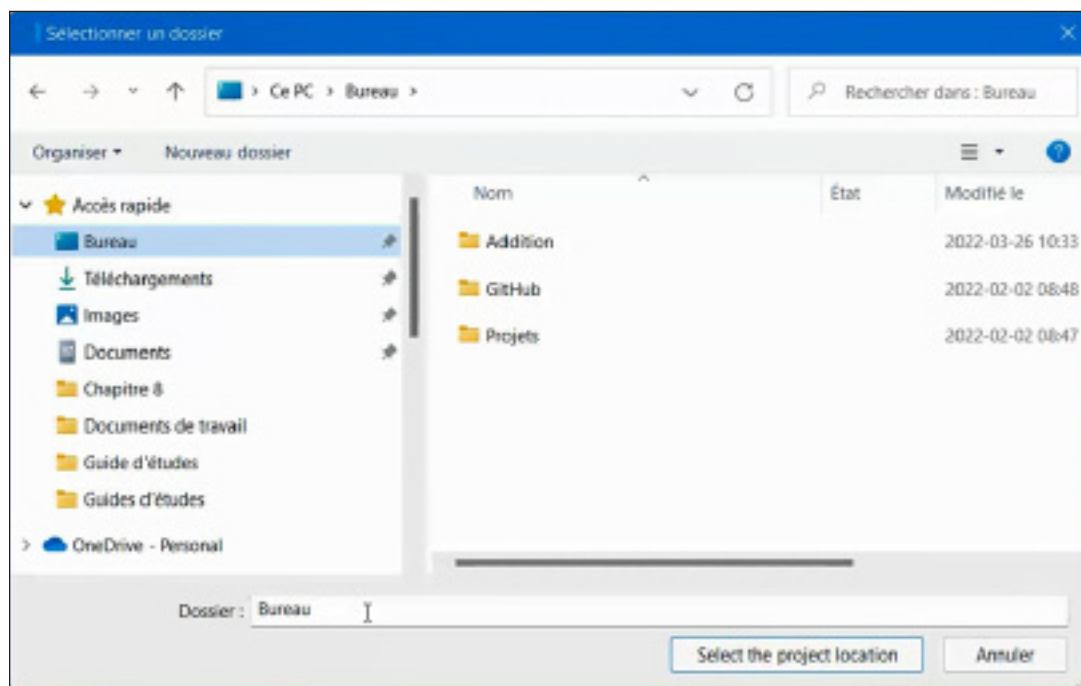
Dans la fenêtre qui s'affiche, sélectionnez l'élément **Java Create Java Project...**



Dans la nouvelle fenêtre qui s'affiche, vous avez le choix entre différents types de projets Java. Pour les besoins de ce cours, sélectionnez l'option **No build tools**.

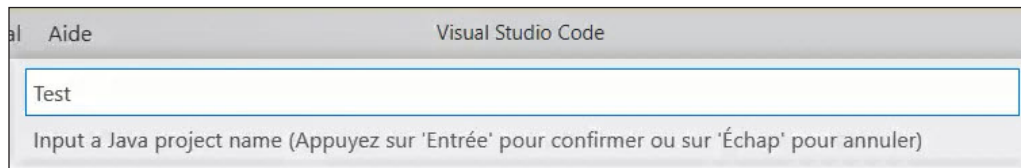


La boîte de dialogue **Sélectionner un dossier** s'affichera à l'écran.

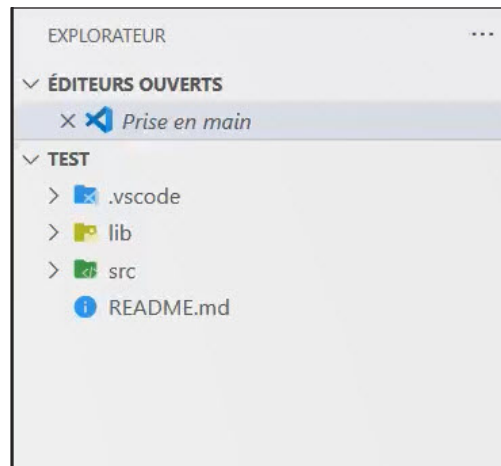
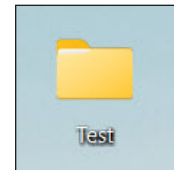


Dans cet exemple, le dossier **Bureau** est sélectionné. Cliquez sur le bouton **Select the project location**.

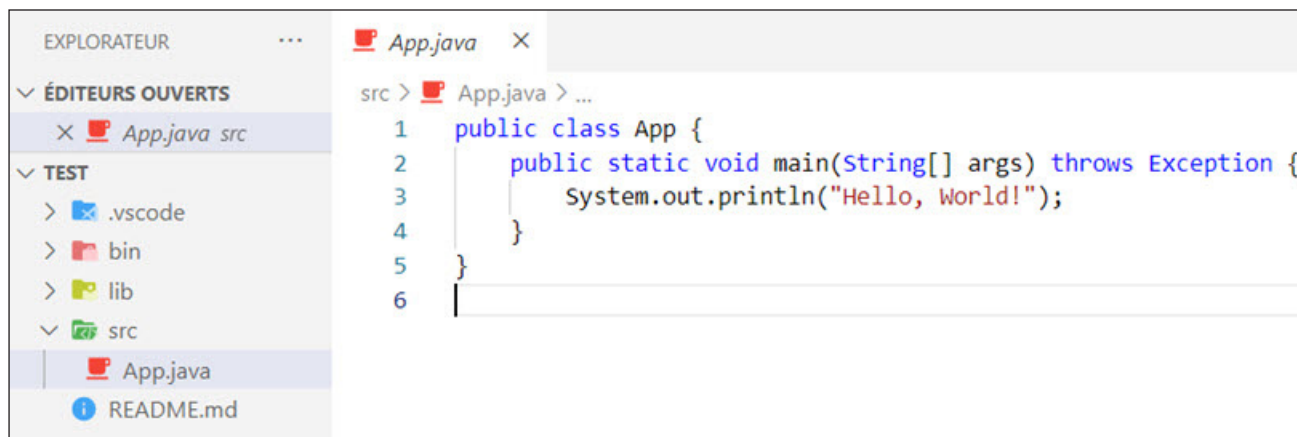
Dans la prochaine zone de saisie qui s'affiche, saisissez le nom de votre projet. Dans cet exemple, nous avons nommé notre projet **Test**. Appuyez sur la touche **Entrée** de votre clavier.



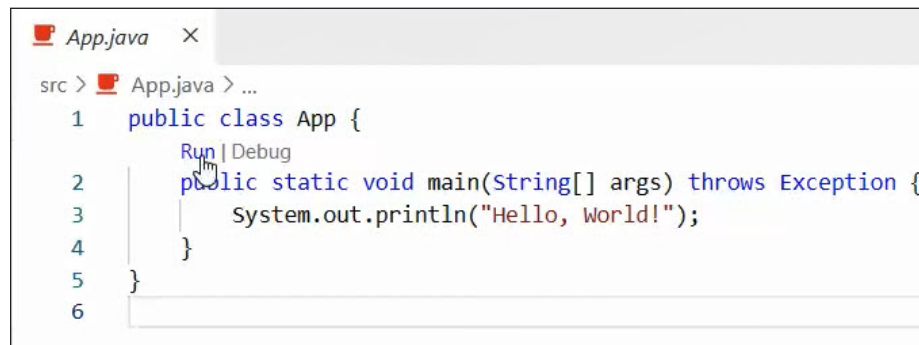
Le dossier racine de votre projet sera créé à l'endroit sélectionné précédemment. Dans notre exemple, un dossier nommé **Test** a été créé sur le **Bureau**. Ce dossier contiendra tous les sous-dossiers et fichiers dont vous aurez besoin pour votre projet et ils seront affichés sous la section **Explorateur** de Visual Studio Code, comme le montre la prochaine figure.



Dans la section Explorateur de Visual Studio Code, développez le dossier **src** afin d'afficher son contenu et cliquez sur l'élément **App.java**. Il s'agit du fichier principal que vous utiliserez pour créer votre application. Le code qui a été créé par défaut s'affichera dans l'**Éditeur de code** à la droite de l'**Explorateur** de VS Code, comme le montre la prochaine figure.



Après quelques secondes, deux éléments cliquables s'afficheront au-dessus de la méthode **main** présente dans ce fichier.

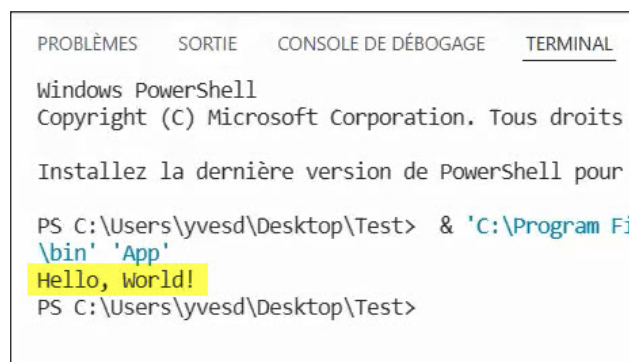


```

src > App.java > ...
1  public class App {
2      public static void main(String[] args) throws Exception {
3          System.out.println("Hello, World!");
4      }
5  }
6

```

Cliquez sur le lien **Run**. Votre fichier sera enregistré si des changements ont été faits et l'application sera compilée puis exécutée dans la fenêtre **Terminal**, sous l'*Éditeur de code*, et le résultat de l'exécution de votre application y sera affiché, comme le montre la prochaine figure.



```

PROBLÈMES  SORTIE  CONSOLE DE DÉBOGAGE  TERMINAL

Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

Installez la dernière version de PowerShell pour Windows.

PS C:\Users\yvesd\Desktop\Test> & 'C:\Program Files\Java\bin' 'App'
Hello, World!
PS C:\Users\yvesd\Desktop\Test>

```

Votre première application en Java est maintenant fonctionnelle. Dans la prochaine leçon et les prochains chapitres, nous allons nous exercer à créer des applications plus complexes en utilisant les algorithmes que nous avons créés dans les leçons précédentes.

Notes

[illegible]