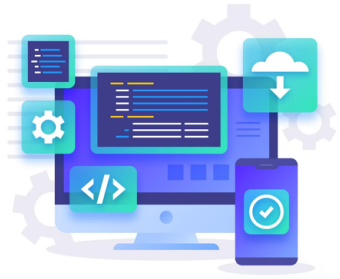	**BTS SIO**	
	Matière	Conception et développement d'applications
	Thème 2	Développement d'applications
	Chapitre	Introduction à Java

Pré-requis	**Compétence professionnelle**	**Savoirs associés**
<u>Bloc 1 : Développer la présence en ligne de l'organisation</u> - Bases de la programmation Web: langage de présentation et de mise en forme, langage d'accès aux données, langage de contrôle - Langage d'interrogation de données <u>Bloc 3 : Cybersécurité</u> - Protéger les données à caractère personnel - Préserver l'identité numérique de l'organisation - Sécuriser les équipements et les usages des utilisateurs	<u>Concevoir et développer une solution applicative</u> - Identifier, développer, utiliser ou adapter des composants logiciels - Exploiter les technologies Web pour mettre en œuvre les échanges entre applications, y compris de mobilité - Utiliser des composants d'accès aux données - Réaliser les tests nécessaires à la validation ou à la mise en production d'éléments adaptés ou développés - Exploiter les fonctionnalités d'un environnement de développement et de tests <u>Gérer les données</u> - Exploiter des données à l'aide d'un langage de requêtes - Développer des fonctionnalités applicatives au sein d'un système de gestion de base de données (relationnel ou non) - Concevoir ou adapter une base de données - Administrer et déployer une base de données	<u>Concevoir et développer une solution applicative</u> - Concepts de la programmation objet: classe, objet, abstraction, interface, héritage, polymorphisme, annotations, patrons de conception, interface de programmation d'applications - Persistance et couche d'accès aux données - Techniques et outils de tests et d'intégration de composants logiciels <u>Gérer les données</u> - Langage et outils de manipulation et d'interrogation d'une base de données

Sommaire

1. Qu'est-ce que Java ?

Java est un langage de programmation très utilisé pour créer des applications sur différents types d'appareils (ordinateurs, téléphones, etc.). Il a été créé par James Gosling et son équipe chez Sun Microsystems en 1995. Aujourd'hui, Java appartient à Oracle Corporation, qui a acquis Sun Microsystems en 2010. C'est un langage orienté objet, ce qui signifie qu'il organise le code en "objets" pour rendre les programmes modulaires et réutilisables.

2. Comment fonctionne Java ?

- **Code Source** : Les développeurs écrivent leur code dans un fichier avec l'extension .java.
- **Compilation** : Java transforme le code source en un langage intermédiaire appelé bytecode.
- **JVM (Java Virtual Machine)** : Ce bytecode est ensuite interprété par la JVM, qui exécute le programme sur l'ordinateur.

3. Pré-requis en tant que développeur

- **Installer le JDK (Java Development Kit)** sur votre ordinateur
- Installer **Extension Pack for Java** dans Visual Studio Code. Cette extension contient :
    - **Language Support for Java™ by Red Hat** : Cette extension, développée par Red Hat, fournit le support de base pour le langage Java dans VS Code.
    - **Debugger for Java** : Ce débogueur permet d'exécuter votre code Java de manière contrôlée pour identifier et corriger les erreurs.
    - **Java Test Runner** : Cet outil permet de tester votre code Java automatiquement.

- **Maven for Java** : Maven est un outil de gestion de projet et de dépendances pour Java.
- **Java Dependency Viewer** : Cette extension permet de visualiser les dépendances et l'organisation des fichiers Java dans votre projet.

4. Premier programme en Java : Hello World

Avant de plonger dans les interfaces graphiques, voyons un exemple simple pour se familiariser avec Java.

HelloWorld.java

```
// Déclaration de la classe nommée HelloWorld
// Le fichier portera le même nom que la classe
public class HelloWorld {
      // Méthode principale, point d'entrée du programme
      public static void main(String[] args) {
         // Affiche "Hello, world!" dans la console
         System.out.println("Hello, world!");
      }
}
```

Explication détaillée du code :

public class HelloWorld {} :

- Déclaration de la classe publique nommée HelloWorld
- "public" signifie que la classe est accessible depuis n'importe quel autre code Java.
- "class" est un mot-clé qui sert à déclarer une nouvelle classe.
- "HelloWorld" est le nom de cette classe, respectant la convention de capitalisation en CamelCase.

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

- Déclaration de la méthode principale : public static void main(String[] args)
- Cette méthode est le point d'entrée du programme, c'est-à-dire que c'est le premier code exécuté lors du lancement du programme.
- "public" signifie que la méthode est accessible depuis n'importe quel autre code Java.
- "static" signifie que la méthode appartient à la classe elle-même et non à une instance (un objet) de la classe.
- "void" indique que la méthode ne retourne aucune valeur.
- "main" est le nom de la méthode, et elle est requise pour démarrer un programme Java.
- "(String[] args)" représente les paramètres de la méthode, un tableau de chaînes de caractères, qui peut contenir les arguments fournis au programme lors de son exécution.

System.out.println("Hello, world!"); :

- Affichage d'un message dans la console.
- "System" est une classe prédéfinie de Java qui contient des objets et des méthodes pour gérer le système d'entrée/sortie.
- "out" est un flux de sortie standard, qui envoie des données vers la console.
- "println" est une méthode qui affiche un message dans la console, suivie d'un retour à la ligne.
- "Hello, world!" est la chaîne de caractères à afficher.

5. Concepts de base en Java

5.1. Variables et types de données

Les variables sont utilisées pour stocker des informations. En Java, chaque variable a un type, comme int pour un nombre entier ou String pour du texte.

```
// Variable de type entier
int nombre = 10;
// Variable de type chaîne de caractères
String message = "Bonjour !";
```

5.2. Les conditions

Comme pour d'autres langages, les conditions "if" permettent d'exécuter un bloc de code sous certaines conditions.

```
int age = 18;
if (age >= 18) {
     System.out.println("Vous êtes majeur.");
} else {
     System.out.println("Vous êtes mineur.");
}
```

5.3. Les boucles

Les boucles sont utilisées pour répéter des instructions plusieurs fois, tant qu'une condition est remplie. Java propose plusieurs types de boucles : for, while, et do-while

5.3.1. La Boucle for

La boucle for est utilisée lorsqu'on sait combien de fois on veut répéter une action.

```
// Boucle qui démarre avec i=0, continue tant que i est inférieur à 5, et
augmente i de 1 à chaque tour
for (int i = 0; i < 5; i++) {
     // Affiche le numéro du tour
     System.out.println("Tour numéro : " + i);
}
```

Explication :

- int i = 0 initialise la variable i à 0
- i < 5 est la condition, la boucle continue tant que i est inférieur à 5.
- i++ augmente i de 1 à chaque tour de boucle.

5.3.2. La Boucle while

La boucle while répète une action tant qu'une condition est vraie. Elle est utile lorsque le nombre de répétitions n'est pas connu à l'avance.

```
// Initialisation de la variable
int i = 0;
// La boucle continue tant que i est inférieur à 5
while (i < 5) {
      // Affiche le numéro du tour
      System.out.println("Tour numéro : " + i);
      // Incrémentation de i pour éviter une boucle infinie
      i++;
}
```

5.3.3. La Boucle do-while

La boucle do-while fonctionne comme while, mais elle s'exécute au moins une fois, même si la condition est fausse dès le départ.

```
// Initialisation de la variable
int i = 0;
do {
        // Affiche le numéro du tour
        System.out.println("Tour numéro : " + i);
         // Incrémente i
         i++;
}
 // La boucle s'arrête quand i atteint 5
while (i < 5);
```

6. Les Interfaces Graphiques avec Swing

Voici un exemple de création pas à pas d'une interface simple. Pour lancer l'interface depuis Visual Studio Code, il faut depuis votre code faire un clic droit et choisir "Run JAVA".

6.1. Créer une Fenêtre Simple

```
// Importation de la bibliothèque Swing pour créer des interfaces graphiques
import javax.swing.*;

// Déclaration de la classe MaPremiereFenetre
public class MaPremiereFenetre {
     // Méthode principale
     public static void main(String[] args) {
        // Création d'une fenêtre avec un titre
        JFrame fenetre = new JFrame("Ma Première Fenêtre");
        // Définit la largeur (400) et la hauteur (300) de la fenêtre
        fenetre.setSize(400, 300);
        // Ferme le programme lorsqu'on ferme la fenêtre
        fenetre.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        // Rend la fenêtre visible à l'écran
        fenetre.setVisible(true);
    }
}
```

6.2. Ajouter des composants : un bouton

```
import javax.swing.*;
// Importation pour les paramètres de couleur et de police
import java.awt.*;

public class MaPremiereFenetre {
      public static void main(String[] args) {
      JFrame fenetre = new JFrame("Ma Première Fenêtre Stylisée");
      fenetre.setSize(400, 300);
      fenetre.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      // Création d'un bouton avec le texte "Cliquez ici"
      JButton bouton = new JButton("Cliquez ici");

      // Stylisation du bouton
      // Couleur bleu vif pour le fond
      bouton.setBackground(new Color(30, 144, 255));
      // Texte en blanc
      bouton.setForeground(Color.WHITE);
      // Police en gras et taille augmentée pour meilleure visibilité
      bouton.setFont(new Font("Arial", Font.BOLD, 18));
      // Supprime le contour de focus
      bouton.setFocusPainted(false);
      bouton.setBorder(BorderFactory.createCompoundBorder(
         // Bordure extérieure grise foncée
         BorderFactory.createLineBorder(Color.DARK_GRAY, 2),
         // Espacement interne pour agrandir le bouton
         BorderFactory.createEmptyBorder(10, 20, 10, 20))
      );
      // Ajout d'une ombre à l'aide d'une marge
      bouton.setMargin(new Insets(5, 5, 5, 5));
      bouton.setOpaque(true);

      // Ajoute le bouton à la fenêtre
      fenetre.add(bouton);

      // Utilisation du layout pour centrer le bouton
      fenetre.setLayout(new FlowLayout());
      fenetre.setVisible(true);
      }
}
```

6.3. Ajouter une action au bouton

```
import javax.swing.*;
import java.awt.*;
// Importation de ActionEvent pour gérer les actions sur les boutons
import java.awt.event.ActionEvent;
// Importation de ActionListener pour écouter les actions
import java.awt.event.ActionListener;

public class MaPremiereFenetre {
     public static void main(String[] args) {
     JFrame fenetre = new JFrame("Ma Première Fenêtre Stylisée");
     fenetre.setSize(400, 300);
     fenetre.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

     JButton bouton = new JButton("Cliquez ici");

     bouton.setBackground(new Color(30, 144, 255));
     bouton.setForeground(Color.WHITE);
     bouton.setFont(new Font("Arial", Font.BOLD, 18));
     bouton.setFocusPainted(false);
     bouton.setBorder(BorderFactory.createCompoundBorder(
        BorderFactory.createLineBorder(Color.DARK_GRAY, 2),
        BorderFactory.createEmptyBorder(10, 20, 10, 20))
     );
     bouton.setMargin(new Insets(5, 5, 5, 5));
     bouton.setOpaque(true);

     fenetre.add(bouton);

     // Action déclenchée lors du clic
     bouton.addActionListener(new ActionListener() {
     @Override
     public void actionPerformed(ActionEvent e) {
     JOptionPane.showMessageDialog(fenetre, "Vous avez cliqué sur le bouton
!");
       }
     });

     fenetre.setLayout(new FlowLayout());
     fenetre.setVisible(true);
     }
}
```

7. Exportation d’un projet avec Visual Studio Code

7.1. Le fichier .jar

Un fichier JAR (Java ARchive) est un fichier compressé qui regroupe plusieurs fichiers Java (classes, images, fichiers de configuration, etc.) en un seul package. Il est utilisé pour :

- Déployer des applications Java : Le JAR contient tout le code et les ressources nécessaires, facilitant le partage et le déploiement d'une application.
- Stocker des bibliothèques Java : Beaucoup de bibliothèques Java sont distribuées sous forme de fichiers JAR que d'autres programmes peuvent importer et utiliser.
- Créer des applications exécutables : Si le fichier JAR contient une classe avec une méthode main(), il peut être configuré pour être "exécutable". Cela signifie qu'on peut lancer l'application directement via le fichier JAR sans compiler le code source à nouveau.

7.2. Les étapes pour créer un exécutable

Dans Visual Studio Code, vous pouvez directement créer un fichier .jar en suivant ces étapes :

- Faites un clic droit “Palette de commande”
- Ecrivez dans le moteur de recherche "Export Jar..."
- Suivez les instructions pour spécifier le point d'entrée (classe main)
- Votre fichier .jar exécutable est maintenant prêt à être utilisé et distribué.

1. Pour aller plus loin

Source Site officiel Java	**Source** Développez.com - Les meilleurs cours et tutoriels pour apprendre JAVA	**Source** Open Classroom - Apprenez à programmer en Java
Lien https://www.java.com/fr	**Lien** https://java.developpez.com/cours	**Lien raccourci** https://tinyurl.com/3xwxnnnu

Synthèse : carte mentale du chapitre

Réalisez la carte mentale du chapitre qui vous servira de synthèse de cours.