Les classes et les objets en JAVA

BAAZIZ Wissal
Mohammed-elarbi El Hattab

Sommaire

Introduction

Java est un langage de programmation polyvalent et largement utilisé, connu pour sa portabilité, sa sécurité, et sa robustesse. L'un des aspects les plus puissants de Java est son adoption complète du paradigme de la Programmation Orientée Objet (POO).

La POO est un modèle de programmation qui organise le code en objets, des unités autonomes qui combinent des données (attributs) et des comportements (méthodes).

Ce modèle permet de créer des logiciels modulaires, réutilisables et faciles à maintenir. En Java, tout est conçu autour des objets et des classes, qui définissent les propriétés et les actions que ces objets peuvent effectuer.

La Classe Main en Java

1. Le Point de Départ d'un Projet Java

La classe Main est généralement le premier point d'entrée dans un projet Java.
Elle contient la méthode main, qui est le point de départ de l'exécution du programme.

```java
public class Main {
    public static void main(String[] args) {
        System.out.println("Bienvenue dans le monde de Java !");
    }
}
```

*public : Un modificateur d'accès qui signifie que la classe Main est accessible depuis n'importe où dans le programme.
*class : Un mot-clé utilisé pour déclarer une classe en Java.
*Main : Le nom de la classe. En Java, il est courant de nommer la classe principale Main, bien que ce ne soit pas obligatoire.

2. Méthode main

```java
public static void main(String[] args) {
    // Corps de la méthode main
}
```

public : Indique que la méthode main est accessible depuis n'importe où dans le programme. Elle doit être publique pour que la JVM (Java Virtual Machine) puisse l'exécuter.
static : Ce mot-clé signifie que la méthode appartient à la classe plutôt qu'à une instance de la classe. Cela permet à la JVM d'exécuter la méthode sans avoir à créer une instance de la classe Main.
void : Indique que la méthode ne retourne aucune valeur.
main : Le nom de la méthode. C'est le point d'entrée du programme où l'exécution commence. C'est une convention en Java, et la JVM recherche cette méthode pour démarrer le programme.
String[] args : Un paramètre qui représente un tableau de chaînes de caractères. Ce tableau peut contenir des arguments de ligne de commande que l'utilisateur peut passer lors de l'exécution du programme.

Les Classes en Java

Qu'est-ce qu'une Classe ?
Une classe est une structure qui permet de modéliser un concept ou un objet du monde réel.
Elle définit des attributs (variables) et des comportements (méthodes) associés à l'objet.

```
public class NomDeClasse {
    // Attributs
    // Méthodes
}
```

Exemple de Classe Simple

```java
public class Voiture {
    String marque;
    String modele;

    void demarrer() {
        System.out.println("La voiture démarre.");
    }
}
```

Membres d'une Classe

Attributs

Variables qui représentent les caractéristiques d'un objet.
Exemples : String marque, int annee.

Constructeurs

Méthodes spéciales utilisées pour initialiser des objets d'une classe.

Méthodes

Fonctions définies dans une classe qui décrivent les comportements ou les actions qu'un objet peut effectuer.
Exemple : void demarrer().

Exemple d'un constructeur :

```java
public Voiture(String marque, String modele) {
    this.marque = marque;
    this.modele = modele;
}
```

Les Objets en Java

Qu'est-ce qu'un Objet en Java ?

- Un objet est une instance d'une classe.
- Chaque objet est une entité concrète qui possède des propriétés (appelées attributs) et des comportements (appelés méthodes) définis par la classe dont il est issu.
- Les objets permettent de représenter des éléments du monde réel dans un programme.

```
ClassName obj = new ClassName();
```

Exemple

Supposons que nous ayons une classe Voiture avec des attributs comme couleur et une méthode pour démarrer le moteur.

```java
public class Voiture {
    String couleur;

    // Méthode pour démarrer le moteur
    void demarrerMoteur() {
        System.out.println("Le moteur est démarré.");
    }
}
```

Création d'un Objet Voiture :

```java
public class Main {
    public static void main(String[] args) {
        // Création d'un objet 'Voiture'
        Voiture maVoiture = new Voiture();

        // Définir la couleur de la voiture
        maVoiture.couleur = "Rouge";

        // Démarrer le moteur de la voiture
        maVoiture.demarrerMoteur();

        // Affichage de la couleur
        System.out.println("Couleur de la voiture : " + maVoiture.couleur);
    }
}
```

Encapsulation en Java

Qu'est-ce que l'Encapsulation ?

- L'encapsulation est un principe fondamental de la Programmation Orientée Objet (POO).
- Elle consiste à protéger les données d'une classe en restreignant l'accès direct à ses attributs.
- Les attributs d'une classe sont généralement déclarés comme privés (private), et leur accès est contrôlé par des méthodes publiques appelées getters et setters.

Pourquoi l'Encapsulation ?

- Sécurité des données : Empêche la modification imprévue des données sensibles.
- Modularité : Facilite la maintenance et les mises à jour du code.
- Contrôle : Permet de contrôler la manière dont les données sont lues ou modifiées.

Exemple

Supposons que nous ayons une classe CompteBancaire qui représente un compte bancaire.

```java
public class CompteBancaire {
    // Attribut privé
    private double solde;

    // Getter pour accéder au solde
    public double getSolde() {
        return solde;
    }

    // Setter pour modifier le solde
    public void deposerArgent(double montant) {
        if (montant > 0) {
            solde += montant;
        }
    }

    // Méthode pour retirer de l'argent
    public void retirerArgent(double montant) {
        if (montant > 0 && montant <= solde) {
            solde -= montant;
        }
    }
}
```

Utilisation de l'Encapsulation :

```java
public class Main {
    public static void main(String[] args) {
        // Création d'un objet 'CompteBancaire'
        CompteBancaire monCompte = new CompteBancaire();

        // Dépôt d'argent
        monCompte.deposerArgent(500);

        // Affichage du solde
        System.out.println("Solde actuel : " + monCompte.getSolde());

        // Retrait d'argent
        monCompte.retirerArgent(200);

        // Affichage du solde après retrait
        System.out.println("Solde après retrait : " + monCompte.getSolde());
    }
}
```

Résultat attendu :

```
Solde actuel : 500.0
Solde après retrait : 300.0
```

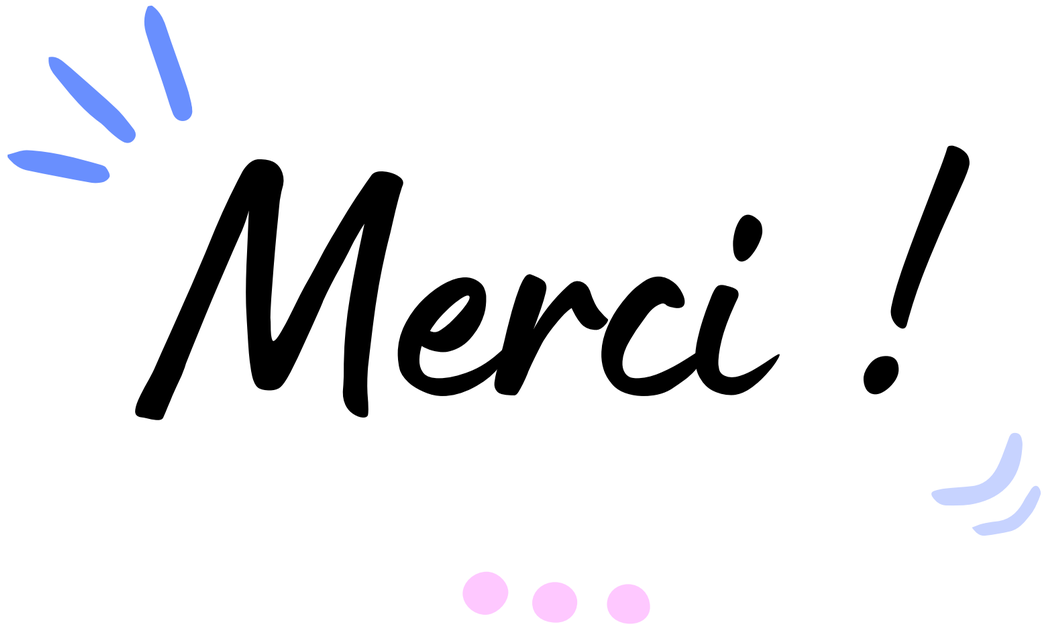
Points Clés sur l'Encapsulation

- Les attributs privés sont protégés et ne peuvent être modifiés directement.
- Les getters et setters permettent un accès contrôlé aux données.
- L'encapsulation améliore la sécurité et la maintenabilité du code.

Conclusion

Cette présentation a exploré les concepts clés de la Programmation Orientée Objet (POO) en Java. Les thèmes abordés incluent la structure de la classe Main, la déclaration et les membres des classes, ainsi que la création et l'utilisation des objets. L'accent a été mis sur l'encapsulation, un principe fondamental qui protège les données et assure un accès contrôlé.

Ces concepts sont essentiels pour développer des applications Java efficaces et maintenables. La compréhension de ces principes permet de créer des logiciels robustes et bien structurés, facilitant ainsi le développement et la gestion des projets en Java.


Merci !