Introduction à la programmation en JAVA

technocité
Centre de compétence

WAVENET
Intelligent IT Solutions

Table des matières

technocité
Centre de compétence

WAVENET
Intelligent IT Solutions

Définition

- Une **classe interne** est une classe définie à l'intérieur d'une autre classe

- Il existe plusieurs types de classes internes :
  - Classe interne
  - Classe interne locale
  - Classe interne anonyme
  - Classe interne statique

- 
Intérêt ? La **classe interne** a accès aux variables et aux méthodes privées de la **classe externe**

Classe interne (1/2)

- Classe définie à l'intérieur d'une autre classe, en dehors de toute méthode

```
public class OuterClass {
    // …
    class InnerClass {
        // …
    }
    // …
}
```

- Une instance d'une classe interne ne peut exister seule, elle est toujours attachée à une instance de la classe englobante

Pour récupérer depuis la classe interne une référence vers la classe externe, la syntaxe est `OuterClass.this`

technocité
Centre de compétence

WAVENET
Intelligent IT Solutions

Classe interne (2/2)

- Il existe plusieurs règles de visibilité pour les classes internes :

Mot clé	Description
-	Accès par défaut : « package », la classe interne sera accessible à l'ensemble des classes situées dans le package
`public`	La classe interne est accessible par toutes les classes
`private`	La classe interne n'est accessible que depuis la classe externe
`protected`	La classe interne est accessible par toutes les classes dérivées de la classe externe, quelque soit leur package

Pour les classes internes publiques, la syntaxe suivante est licite

Classe interne locale (1/3)

- Une classe interne peut être déclarée dans une méthode, on parle alors de classe interne locale

```
public class OuterClass {
    // …
    public void outerMethod() {
       class TimePrinter implements ActionListener {
          public void actionPerformed(ActionEvent event) {
                 Date now = new Date();
                 System.out.println("Time : " + now);
          }
       }

       ActionListener listener = new TimePrinter();
       Timer t = new Timer(interval, listener);
       t.start();
    }
    // …
}
```

technocité
Centre de compétence

WAVENET
Intelligent IT Solutions

Classe interne locale (2/3)

- Les classes internes locales ne sont jamais déclarées avec une spécification d'accès. Leur visibilité est toujours limitée au bloc de code dans lequel elles sont déclarées

- Les classes internes locales peuvent non seulement accéder aux champs de la classe externe, mais également aux variables locales déclarées final …

Classe interne locale (3/3)

- Pour quelles raisons seules les variables locales final sont accessibles ?

Au moment de la compilation, la classe interne est extraite de la classe externe. Une classe régulière est générée à partir de la classe interne.

Le compilateur y ajoute un attribut du type de la classe externe ainsi que des attributs correspondants aux variables locales finales.

A l'instanciation de la classe, la référence vers la classe externe ainsi que des copies des variables finales sont passées en argument au constructeur.

Classe interne statique (1/2)

- Une classe interne statique ne dispose pas de référence vers la classe englobante

```
public class OuterClass {
    // …
    public static class InnerClass {
        //…
    }
    // …
}
```

- Elle n'a donc pas besoin d'un objet de la classe englobante pour exister

Classe interne statique (2/2)

- Pour instancier une classe interne statique

```
OuterClass.StaticClass = new OuterClass.StaticClass();
```

- Les règles de visibilité autorisées sont les mêmes que pour les classes internes non statiques : « package », public, private, protected

Classe interne anonyme (1/3)

- Une **classe interne anonyme**
  - est une classe sans nom, déclarée au cours d'une assignation ou comme argument d'une méthode
  - étend une classe abstraite ou implémente une interface

```
public class OuterClass {
    // …
    public void outerMethod() {
        ActionListener listener = new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                    Date now = new Date();
                    System.out.println("Time : " + now);
            }
        };

        Timer t = new Timer(interval, listener);
        t.start();
     }
    // …
}
```

technocité
Centre de compétence

WAVENET
Intelligent IT Solutions

Classe interne anonyme (2/3)

- Une classe interne anonyme n'a pas de constructeur, mais peut avoir un bloc d'initialisation non statique

```
ActionListener listener = new ActionListener() {
    {
        //…
    }

    public void actionPerformed(ActionEvent event) {
        //…
    }
}
```

- Les blocs d'initialisation apparaissant dans une classe sont exécutés avant le constructeur

technocité
Centre de compétence

WAVENET
Intelligent IT Solutions

Classe interne anonyme (3/3)

- Comme pour les classes internes locales, les paramètres utilisés à l'intérieur d'une classe interne anonyme doivent être déclarés final

- Un des avantages des classes internes anonymes est de fournir un moyen commode pour implémenter une méthode callback, lancer un thread ou encore implémenter un listener

Exercices

1. Créer un compte en banque courant qui permet d'ajouter et retirer de l'argent. Le compte doit se souvenir de la dernière opération, utilisez une classe interne pour y arriver.

2. Modifiez l'exercice précédent en rajoutant un compte d'épargne, celui-ci doit se rappeler de ses deux dernières opérations.

3. Créer une classe *WeatherDataGenerator* qui génère toutes les 5 secondes un nouvel échantillon de température et d'humidité. Les valeurs aléatoires doivent être dans des intervalles réalistes. L'interface *TimerTask* est imposée.

4. Il est possible pour le compte épargne d'afficher ses deux dernières opérations, pour ce faire passez par la méthode *forEach()* de l'interface *Collection*.

Exercices

5. Reprenez l'exemple sur les villes et pays. Triez les villes par ordre alphabétique ne passez plus par une classe qui implémente Comparator, mais par une classe anonyme.