Classes internes

Alexis Nasr (d'après les slides de Arnaud Labourel)

Classes internes

- Il est possible de définir une classe à l'intérieur d'une autre (classe interne ou classe imbriquée ou *nested class*)
- La syntaxe est la suivante :

```
class ClasseExterne
{
...
    class ClasseInterne
    {
        ...
    }
}
```

- Cela permet de regrouper des classes qui ne sont utilisées que dans le cadre d'une autre classe.
- On distingue deux types de classes internes :
  - Les classes internes normales (non statiques)
  - Les classes internes statiques

Classes internes non statiques

Exemple

```
public class LinkedList {
    private Node first = null;
    public void add(String data) {
        first = new Node(data, first);
    }
    class Node {
        private String data;
        private Node next;
        public Node(String data, Node next) {
            this.data = data;
            this.next = next;
        }
    }
}
```

Exemple

- Les instances de la classe *Node* ne sont accessibles qu'à travers une instance de *LinkedList*

```
public class TestLinkedList {
    public static void main(String args[]) {
        // on ne peut pas faire ça :
        Node node = new Node("a", null);
        //mais on peut faire ça :
        LinkedList l1 = new LinkedList();
        LinkedList.Node n = l1.new Node("a", null);
    }
}
```

Accès aux membres de la classe externe

- Une instance de classe interne a accès aux membres (même privés) de sa classe externe.

```
public class LinkedList {
    private Node first = null;
    public void add(String data) {
        first = new Node(data, first);
    }
    class Node {
        private String data;
        private Node next;
        public Node(String data, Node next) {  ... }

        private boolean isFirst() {
           return this==/*référenceVersLinkedList.*/first;
        }
    }
}
```

Classe interne

Il est possible d'utiliser la méthode `isFirst` dans `LinkedList` :

```
public class LinkedList {
  /* Code de la classe interne statique Node
     et champs et méthodes de la classe LinkedList. */
  public void print() {
    Node node = first;
    while (node!=null) {
      System.out.print("["+node.data
          +","+node.isFirst()+"]");
      node = node.next;
    }
  }
}
```

Classe interne

Exemple d'utilisation de la classe précédente :

```
LinkedList list = new LinkedList();
list.add("a");
list.add("b");
list.add("c");
list.print();
```

Cet exemple produit la sortie suivante :

```
[c,true][b,false][a,false]
```

Accès aux membres de la classe interne

- Une classe interne est un membre de sa classe externe
- Son accès peut être contrôlé à l'aide des modificateurs
  - `private`
  - `protected`
  - `public`.

Classes internes statiques

Classes internes statiques

- Dans le cas d'une classe interne (non statique), une instance de la classe interne ne peut exister sans qu'il n'existe une instance de la classe externe.
- Dans le cas d'une classe interne statique, une instance de cette dernière peut exister, sans qu'il n'existe une instance de la classe externe.
- Une classe interne statique peut accéder aux membres statiques de sa classe externe
- Mais elle ne peut accéder aux membres non statiques de sa classe externe, elle ne peut y accéder qu'à travers une instance de sa classe externe.
- Les membres statiques de la classe externe peuvent accéder à la classe interne.

Exemple

```
public class LinkedList {
    public static class Node {
        private String data; private Node next;
        public Node(String data, Node next) {
            this.data = data; this.next = next;
        }
    }
}
```

Il est possible d'instancier la classe interne sans qu'une instance de `LinkedList` existe car elle est statique :

```
LinkedList.Node node = new LinkedList.Node("a", null);
```

Classe interne statique

Il est également possible de la rendre privée à la classe `LinkedList` :

```
public class LinkedList {
    private static class Node {
        private String data;
        private /*LinkedList.*/Node next;
        public Node(String data, Node next) {
            this.data = data;
            this.next = next;
        }
    }
}
```

- Dans ce cas, seules les méthodes de `LinkedList` pourront l'utiliser.
- Les méthodes statiques définies dans `LinkedList` peuvent également utiliser cette classe interne car elle est statique.

Classe interne statique

Une classe interne statique ne peut accéder qu'aux attributs et méthodes statiques de la classe qui la contient :

```
public class LinkedList {
    private static class Node {
        /* Champs et méthodes de Node. */
        boolean isFirst() {
            return this==first; // interdit !
        }
    }
    private Node first;
    /* Autres champs et méthodes de LinkedList. */
}
```

Classe interne statique

Exemple d'implémentation de méthodes dans la classe `LinkedList` :

```
public class LinkedList {
    /* Code de la classe interne statique Node. */
    private Node first = null;
    public void add(String data) {
        first = new Node(data, first);
    }
    public void print() {
        Node node = first;
        while (node!=null) {
            System.out.print("["+node.data+"]");
            node = node.next;
        }
    }
}
```

Classe interne statique

Exemple d'utilisation de la classe précédente :

```
LinkedList list = new LinkedList();
list.add("a");
list.add("b");
list.add("c");
list.print();
```

Cet exemple produit la sortie suivante :

```
[c][b][a]
```