Les blocs d'instructions

Les blocs

En Java, les instructions peuvent être regroupées en **blocs**.

Les blocs sont identifiés par des délimiteurs de début et de fin : **{** et **}**

Exemple:

```
{
  Scanner keyb = new Scanner(System.in);
  int i;
  double x;

  System.out.println("Valeurs pour i et x : ") ;
  i = keyb.nextInt();
  x = keyb.nextDouble();
  System.out.println("Vous avez saisi : i = " + i +
                     ", x = " + x);
}
```

Les blocs

Les blocs ont en Java une grande autonomie.

Ils peuvent contenir leurs propres déclarations et initialisation de variables:

```
if (i != 0) {
  int j = 0;

  ...
  j = 2 * i;
  ...
}
// A partir d'ici, on ne peut plus utiliser j
```

Notion de portée

- Les variables déclarées à l'intérieur d'un bloc sont appelées **variables locales** (au bloc). Elles ne sont accessibles qu'à l'intérieur du bloc.

```
if (i != 0) {
  int j = 0;

  ...
  j = 2 * i;
  ...
}
// A partir d'ici, on ne peut plus utiliser j
```

Notion de portée

- Les variables déclarées à l'intérieur d'un bloc sont appelées **variables locales** (au bloc). Elles ne sont accessibles qu'à l'intérieur du bloc.

- Les variables déclarées en dehors de `main` sont de portée **globales** (à la classe).
  Elles sont accessibles dans toute la classe.

Pour ces variables, on distingue en Java des variables de classes et des variables d'instance.

Notion de portée

- Les variables déclarées à l'intérieur d'un bloc sont appelées **variables locales** (au bloc). Elles ne sont accessibles qu'à l'intérieur du bloc.

- Les variables déclarées en dehors de `main` sont de portée **globales** (à la classe).
  Elles sont accessibles dans toute la classe.

Bonne pratique: Déclarer les variables au plus près de leur utilisation.

Notion de portée

Déclarer les variables au plus près de leur utilisation

Par exemple, si la variable `j` n'est pas utilisée après la condition,

écrivez:

```
if (i != 0) {
  int j = 0;

  ...
  j = 2 * i;
  ...
}
```

plutôt que:

```
int j = 0;
if (i != 0) {

  ...
  j = 2 * i;
  ...
}
```

Notion de portée

La **portée** d'une variable, c'est l'ensemble des lignes de code où cette variable est accessible, autrement dit où elle est définie, existe, a un sens.

```
if (i != 0) {
  int j = 0;

  ...
  j = 2 * i;
  ...
  if (j != 2) {
    int k = 0;

    ...
    k = 3 * i;
    ...
  }
  ...
}
```

Portée : règle

```
if (i != 0) {
  int j = 0;

  ...
  j = 2 * i;
  ...
  if (j != 2) {
    int j = 0; // interdit

    ...
    j = 3 * i;
    ...
  }
  ...
}
```

En Java, on ne peut pas utiliser le nom d'une variable déclarée plus globalement pour déclarer une autre variable.

Cela permet d'éviter des ambiguïtés entre noms de variables.

Portée : cas des itérations

La déclaration d'une variable à l'intérieur d'une itération est une déclaration **locale au bloc de la boucle**, et aux deux instructions de test et d'incrément:

```
for(int i = 0; i < 5; ++i) {
  System.out.println(i);
}
// A partir d'ici, on ne peut plus utiliser ce i
```