Support de cours

Cours:

# Initiation à la programmation (en Java)

Vidéo:

## Init-JAVA-03-5-Blocs-pt3

Concepts (extraits des sous-titres générés automatiquement) :

**Cas particulier de portée. Portée d'une variable. Variable j. Bloc d'instruction. Ensemble des lignes de code. Boucle for. Première instruction if. Variables de même nom. Erreur de compilateur de java. Cas de la portée de variables. Bloc d'instructions. Notion d'endroit. Deuxième instruction if. Boucle itérative. Genre de choses.**

vers la recherche de séquences vidéo
(dans Initiation à la programmation (en Java).)

vers la vidéo

EPFL

# Fonctions : blocs
## (Partie 3)

Initiation à la programmation (Java)

Jamila Sam, Vincent Lepetit et Jean-Cédric Chappelier

...

notes

résumé

0m 0s

# 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;
        ...
    }
    ...
}
```

Cette notion d'endroit où l'on utilise la variable, plus exactement d'endroit où l'on peut utiliser une variable, c'est ce que l'on appelle la portée. La portée d'une variable c'est l'ensemble des lignes de code où cette variable est accessible, utilisable, où on peut en parler. Si je prends un exemple, donc avec ici une première instruction if qui a ici son bloc, donc qui contrôle ici un bloc d'instruction, et puis une deuxième instruction if, ici, qui

notes

résumé

0m 1s

# 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;
        ...
    }
    ...
}
```

contrôle elle-même aussi son bloc d'instructions. Dans le bloc de plus haut niveau, on va déclarer ici, une variable j, et puis dans le bloc le plus profond on a déclaré une variable k, que l'on utilise. La portée de k, c'est-à-dire l'ensemble des lignes de code où k est défini, ça va être le bloc ici de la deuxième instruction if.

notes

résumé

0m 37s

# 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.

Alors que la portée de j, c'est-à-dire l'ensemble des lignes dans lesquelles j est défini, utilisable, ça va être le bloc ici dans lequel il a été défini. Donc vous voyez ici, k et j ont des portées différentes, indiquées par ces deux blocs. Une question qu'on pourrait se poser, c'est: et qu'est-ce qui se passe si j'avais ici déclaré une variable j dans ce bloc? Bien, en Java on n'a pas le droit de faire ce genre de choses, on n'a pas le droit d'avoir deux variables de même nom dans une portée, dans une même portée.

notes

résumé

1m 1s


Donc par exemple, si j'avais ici déclaré dans le deuxième bloc, une variable j, je vais avoir une erreur de compilateur de Java qui va me dire que c'est interdit, qu'il y a ambiguïté de noms. Donc en Java, ceci est interdit. Vous ne pouvez pas avoir dans une même portée, deux objets différents qui ont le même nom. Mais ceci est possible dans d'autres langages de programmation. Un cas particulier de portée à bien comprendre, à connaître, c'est le

notes

résumé

1m 37s

# 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:

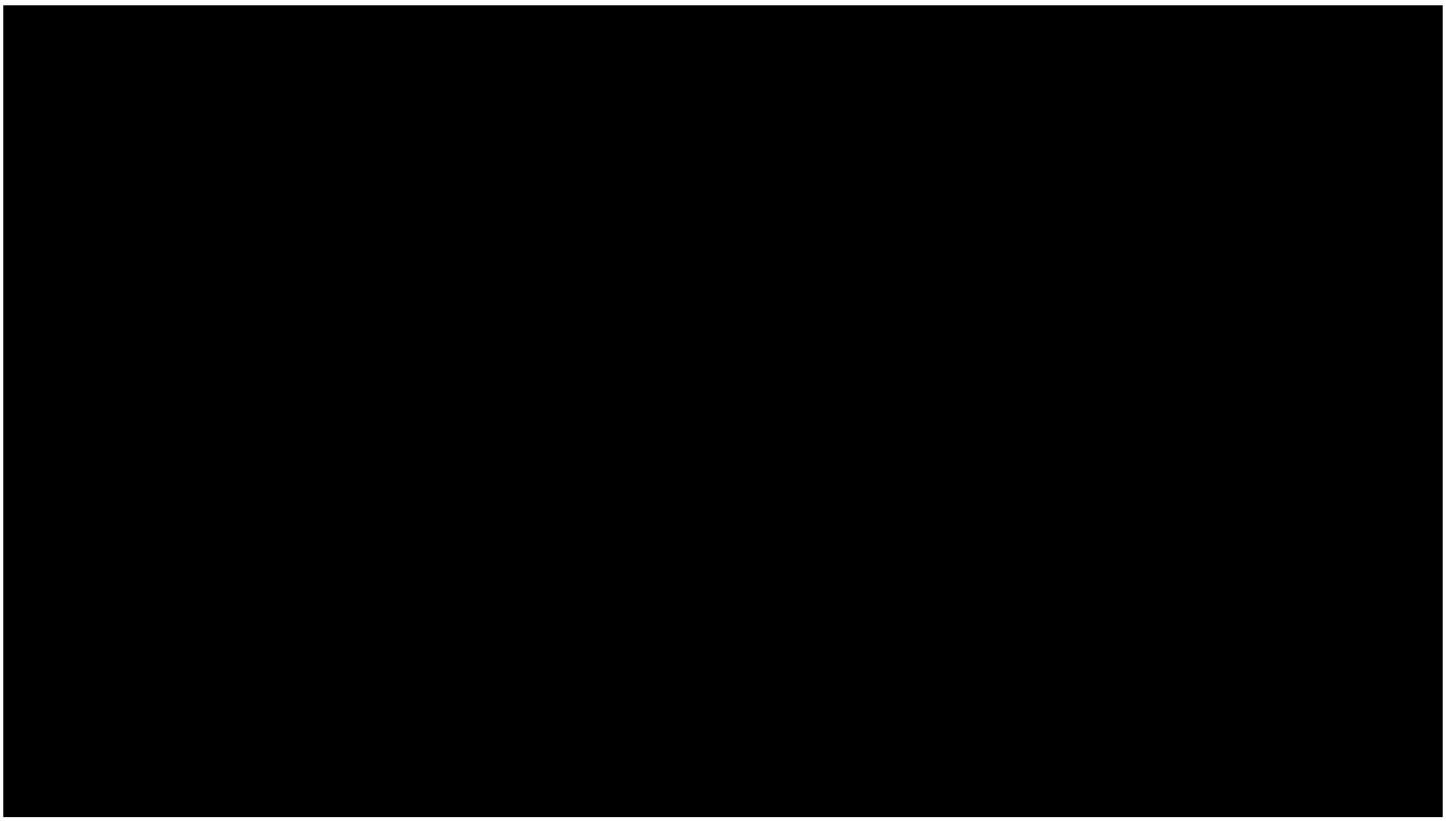
```
for(int i = 0; i < 5; ++i) {
  System.out.println(i);
}
// A partir d'ici, on ne peut plus utiliser ce i
```

cas de la portée de variables de boucle itérative, de boucle for. Si je prends donc un exemple ici, avec une boucle for qui déclare une variable i de type entier, que je peux bien sûr donc utiliser dans le bloc, la portée de ce i va être donc, le bloc contrôlé par l'instruction for, ainsi évidemment que la partie

notes

résumé

2m 1s



condition et la partie incrément de la boucle for, mais cette variable i va être, donc, locale au for, locale, à la boucle itérative. À partir de la fin du bloc contrôlé par le for, on ne pourra plus utiliser cette variable i, elle est locale à l'itération for. i, elle est locale à l'itération for.

notes

résumé

2m 25s