Support de cours

Cours:

Initiation à la programmation (en Java)

Vidéo:

Init-JAVA-04-3-tableaux-usage-pt1

Concepts (extraits des sous-titres générés automatiquement) :

Tableau de taille fixe. Tels tableaux. Ensemble de valeurs. Nom de variable. Type des valeurs du tableau. Fait val. Manipulations classiques. Affectation de ce type. Éléments du tableau. Première manipulation. Taille du tableau. Situation suivante. Solutions classiques. Notation particulière. Moment de la construction du tableau.

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

vers la vidéo

EPFL

Tableaux : traitements courants
(Partie 1)

Initiation à la programmation (Java)

Jamila Sam, Vincent Lepetit et Jean-Cédric Chappelier

...

notes

résumé

0m 0s

Affichage d'un tableaux de taille fixe

Le code suivant :

```
double[] t1 = {1.1, 2.2, 3.4};
System.out.println(t1);
```

affiche la référence au tableau `t1`, donc une adresse.

Si l'on veut faire afficher les entrées du tableau référencé par `t1`, il faut prévoir une boucle (itération) !

Nous savons à ce stade déclarer et initialiser un tableau de taille fixe Nous allons nous intéresser aux diverses manipulations classiques qu'on peut faire sur de tels tableaux. Première manipulation : l'affichage. Imaginons par exemple, que nous ayons déclaré et initialisé un tableau t1 de cette façon-là. Nous allons aboutir à la situation suivante en mémoire, Dans la variable t1, nous avons une référence, une adresse vers le tableau. Le tableau est initialisé, ici, de cette façon. Supposons maintenant que je souhaite afficher le tableau en utilisant simplement

notes

résumé

0m 1s

```java
public class IterationTableau
{
    public static void main(String[] args)
    {
        int[] tab = {1,2,3};

        // iteration sur ensemble de valeurs
        for(int val : tab){
          System.out.println(val);
        }

        // iteration classique
       for(int i = 0; i < 3; ++i){
         System.out.println(tab[i]);
       }
    }
}
```

une instruction telle que celle-ci. Je passe à l'instruction System.out.println, le tableau t1 que je souhaite faire afficher. Ici, ce que je vais afficher, c'est le contenu de t1 qui se trouve être une référence. Ceci va se traduire par un affichage un peu étrange qui aura cette allure-là. Quelque chose comme fe25235d qui va correspondre à quelque chose qui est une référence vers le tableau, quelque chose de pas très compréhensible. En réalité, ce que l'on souhaite faire, c'est plutôt afficher le contenu du tableau. Pour ceci, nous n'avons pas d'autre choix que de prévoir une boucle pour itérer sur chacun des éléments du tableau et les afficher un par un. Supposons que nous souhaitions faire afficher chacun des éléments d'un tableau. Imaginons que ce tableau ait, au préalable, été déclaré de cette façon.

notes

résumé

0m 37s

```java
public class IterationTableau
{
    public static void main(String[] args)
    {
        int[] tab = {1,2,3};

        // iteration sur ensemble de valeurs
        for(int val : tab){
          System.out.println(val);
        }

        // iteration classique
       for(int i = 0; i < 3; ++i){
         System.out.println(tab[i]);
       }
    }
}
```

Il s'agit ici d'un tableau d'entiers. Comment itérer sur un tel tableau pour afficher chacune des cases ici ? Eh bien, nous disposons de deux solutions classiques. La première est ce que l'on appelle "itération sur ensemble de valeurs" qui va s'écrire de cette façon. Le mot réservé for et entre parenthèses, il faudra indiquer le type des valeurs du tableau, ce qui doit être évidemment du même type que ce que l'on a spécifié ici. Ensuite nous devons donner un nom de variable qui va permettre de désigner une des cases du tableau. Ensuite, après deux points, nous devons indiquer le tableau sur lequel nous voulons itérer. Ensuite, il suffit par exemple pour faire afficher la case courante, d'écrire ce genre d'instruction. En fait val va prendre tour à tour chacun des éléments du tableau et nous allons donc les afficher en séquences les uns après les autres au moyen de cette itération sur l'ensemble des valeurs. La solution encore plus classique consiste à utiliser un for tel que nous les connaissons jusqu'ici.

notes

résumé

1m 25s

```java
public class IterationTableau
{
    public static void main(String[] args)
    {
        int[] tab = {1,2,3};

        // iteration sur ensemble de valeurs
        for(int val : tab){
          System.out.println(val);
        }

        // iteration classique
       for(int i = 0; i < 3; ++i){
         System.out.println(tab[i]);
       }
    }
}
```

Un for qui utilise un indice, qui va se déplacer de 0 jusqu'à la taille du tableau. Ici, cette façon de spécifier la taille n'est pas bonne parce qu'elle est décorrélée du tableau lui-même. On a mis en dur la valeur 3. Va se poser à nous, la question de comment spécifier cette taille un peu plus proprement. Nous allons le voir plus tard.

notes

résumé

2m 29s

Accès aux éléments d'un tableau (1)

Très souvent, on voudra accéder aux éléments d'un tableau en effectuant une *itération* sur ce tableau.

Il existe en fait au moins *trois* façons d'itérer sur un tableau :

- avec les itérations sur ensemble de valeurs

```
for(Type element : tableau)
```

  ☞ *Type* est le type des éléments du tableau

- avec une itération `for` « classique » :

```
for(int i=0; i < TAILLE; ++i)
```

  ☞ *TAILLE* ? ? voir plus loin

- avec des itérateurs (non présenté dans ce cours)

Donc on avance sur notre indice pour itérer sur chacune des cases du tableau. Nous accédons aux cases du tableau au moyen du mécanisme d'indexation. Pour résumer, si nous souhaitons itérer sur chacune des valeurs d'un tableau nous disposons d'outils tel que les itérations sur ensemble de valeurs. Nous venons d'en voir un exemple concret juste précédemment. Nous pouvons également utiliser une itération for classique. A ce moment-là, ce pose à nous le problème du calcul de comment retrouver la taille du tableau, ce que nous allons voir juste un petit peu plus loin. Il existe en réalité une troisième façon d'itérer sur un tableau, celle qui utilise la notion d'itérateur.

notes

résumé

2m 49s

Nombre d'éléments d'un tableau

Pour connaître la **taille d'un tableau** :

- `nomTableau.length`

Exemple :

```
int[] scores = {1000, 1500, 2490, 6450};
System.out.println(scores.length); // 4
boolean[] bs = {true, false};
System.out.println(bs.length);     // 2
```

Attention ! `length` donne le **nombre possible** d'éléments. Le remplissage effectif du tableau n'a pas d'importance !

Exemple :

```
int[] scores = new int[2];
System.out.println(scores.length); // 2
```

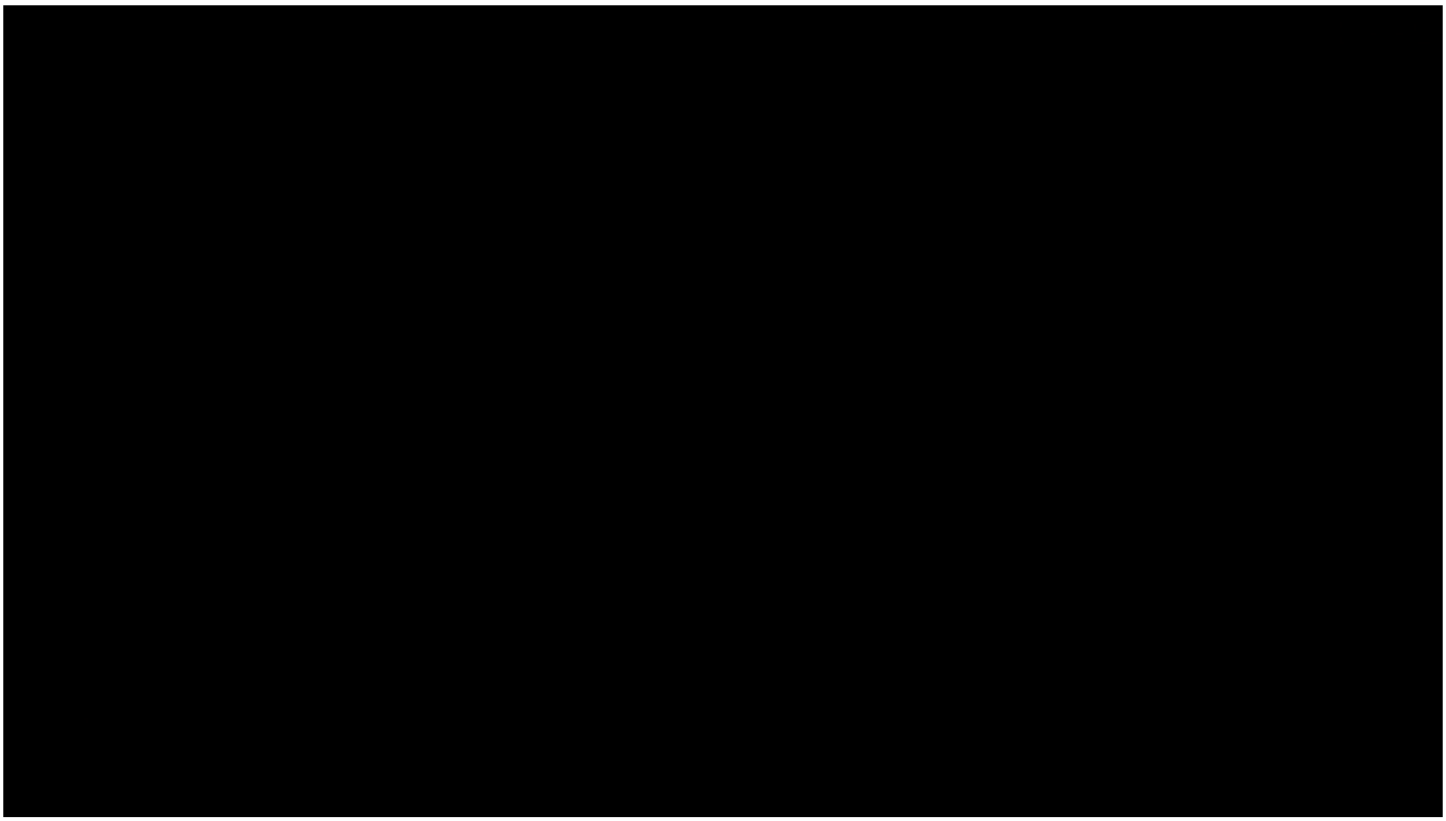
scores[0] = 100;
= {100, 200};

Mais cette notion n'est pas présentée dans le cadre de ce cours. Revenons à la question qui nous préoccupe pour les itérations classiques : comment spécifier la taille d'un tableau de taille fixe ? Comment retrouver cette taille ? En Java, il est possible de connaître la taille d'un tableau à partir de son nom. Il faut pour cela avoir recours à une notation particulière. Cette notation est la suivante : j'utilise le nom du tableau, suivi de point, suivi du mot réservé length. Un exemple concret : je déclare/initialise un tableau scores contenant quatre éléments. Si je veux faire afficher la taille du tableau, il me suffit d'avoir recours à la notation que je viens de décrire. Donc nom du tableau, suivi de "point", suivi de length qui ici va, bien évidemment, afficher 4. Un exemple analogue ici, avec un tableau de booléen à deux entrées. Il est important de noter ici que length va vous donner toujours le nombre possible d'éléments et que le remplissage effectif n'a pas d'importance. En clair, le fait que vous ayez explicitement mis des valeurs ou pas n'est pas pris en compte. Ici, un exemple concret: Ici, nous déclarons un tableau sans l'initialiser avec des valeurs concrètes, effectives. Ni par une affectation de ce type, ni par l'affectation d'un tableau global avec un certain nombre d'éléments dedans. Je n'ai pas mis de valeurs explicites à l'intérieur de mon tableau.

notes

résumé

3m 25s



Pourtant, si je fait afficher la longueur du tableau, je retrouve effectivement 2. Pourquoi ? Parce qu'ici, au moment de la construction du tableau, je lui ai alloué une taille de 2. Ce qui signifie que je vais avoir cette situation en mémoire. La variable score est une référence vers un tableau à deux entrées. Et l'on sait que par défaut, ces entrées sont initialisées avec 0 pour les entiers. Donc, effectivement, nous avons bel et bien affaire à un tableau de taille fixe, contenant deux entrées. Length va nous donner cette taille. Peu importe que nous l'ayons initialisé avec des valeurs explicites ou pas. avec des valeurs explicites ou pas.

notes

résumé

4m 49s