Support de cours

Cours:

Initiation à la programmation (en Java)

Vidéo:

Init-JAVA-04-2-tableaux-declaration-pt2

Concepts (extraits des sous-titres générés automatiquement) :

Données de types évolués. Données de types de base. Notion de tableaux de taille. Chaînes de caractères. Données de types. Exemple des tableaux. Variable de type de base. Type élémentaire. Situation mémoire. Terme de jargon. Moyen de références. Donnée de type évolué. Exemple. Grande incidence. Objet de type composé.

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

vers la vidéo

EPFL

Tableaux : déclaration
(Partie 2)

Initiation à la programmation (Java)

Jamila Sam, Vincent Lepetit et Jean-Cédric Chappelier

...

notes

résumé

0m 0s

Initialisation d'un tableau de taille fixe (1)

EPFL

Si l'on connaît les valeurs de tous les éléments lors de la déclaration du tableau

☞ une seule instruction de **déclaration-initialisation**

int[] scores = {1000, 1500, };

1. Déclarer le type du tableau
2. Indiquer les éléments entre accolades
3. Séparer les éléments par des virgules

Voyons maintenant à quelle situation mémoire cela correspond.

notes

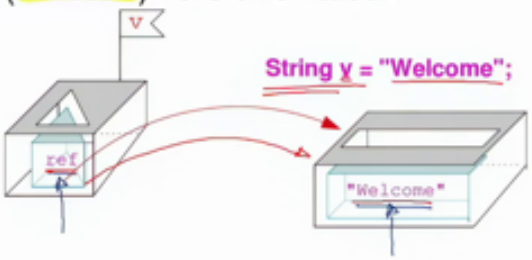
résumé

0m 1s

Types de base et types évolués

- Toute variable de type de base stocke directement une valeur :

- Toute variable de type évolué, comme les tableaux ou les chaînes de caractères (`String`) que vous allez voir dans ce cours, stocke une référence (adresse) vers une valeur :



$v_1 = v_2$

$v_1 == v_2$

Avant d'étudier en détails la notion de tableaux de taille fixe en Java, et puisque nous commençons à manipuler des données de types évolués, il est important de comprendre qu'en Java, les données de types de base et les données de types composés, évolués, ne sont pas stockées en mémoire de la même façon. Par exemple, ici si je déclare une variable « x » de type élémentaire, il faut savoir que lorsque je manipule une variable de type de base et bien cette variable va directement stocker la valeur. Ce qu'on peut voir ici dans ce petit schéma, la variable « x » stocke directement la valeur qui lui est associée. Ceci n'est pas le cas quand je commence à manipuler des données de types évolués. Les données de types évolués, comme par exemple des tableaux ou des chaînes de caractères, que nous allons voir dans des séquences ultérieures sont en fait stockées au moyen de références, d'indirections, des adresses. Par exemple, si je déclare une variable de type chaîne de caractères, nous allons voir qu'il existe un type pour cela qui s'appelle string, alors la situation en mémoire est la suivante : la variable « v » ne stocke pas directement la valeur chaîne de caractères associée, mais stocke une référence, une adresse, vers la chaîne de caractères en question. Et ceci a une grande incidence sur la sémantique de l'affectation en Java. Par exemple, imaginez que j'affecte une variable « v2 » à une variable « v1 », qu'est-ce que cela signifie maintenant concrètement ? Suis-je en train de modifier la référence stockée dans « v1 » ? Ou suis-je en train de modifier l'objet qui est référencé par « v1 » ? De la même façon, la sémantique de l'opérateur « == » est également impactée. Que veut dire comparer « v1 » et « v2 » ? Suis-je en train de comparer des références ?

notes

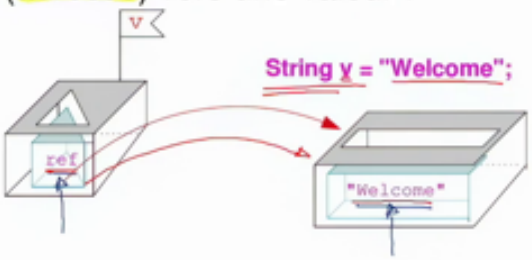
résumé

0m 6s

Types de base et types évolués

- Toute variable de type de base stocke directement une valeur :

- Toute variable de type évolué, comme les tableaux ou les chaînes de caractères (`String`) que vous allez voir dans ce cours, stocke une référence (adresse) vers une valeur :



$v_1 = v_2$

$v_1 == v_2$

Suis-je en train de comparer les chaînes pointées ?

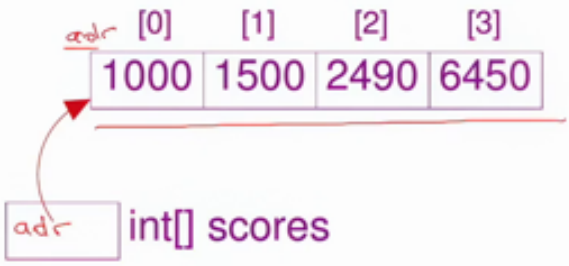
notes

résumé

Situation en mémoire

EPFL

Important : Un tableau n'est pas de type de base,
il est donc manipulé via une **référence** !



int[] Scores
= {1000, 1500, 2490, 6450};

Nous allons voir aussi qu'il existe une incidence par rapport à l'affichage. Si par exemple j'affiche un objet de type composé. Que suis-je en train d'afficher ? Une adresse ? Ou l'objet référencé ? Donc toutes ces questions vont se poser lorsque je manipule des objets de types composés dans un programme Java. En Java, un tableau est une donnée de type évolué, il est donc manipulé au travers d'une indirection, d'une référence. Donc par exemple si je déclare-initialise une variable « score » de type tableau d'entiers de taille fixe de cette façon là. La situation en mémoire va être la suivante la variable « score » ne contient pas directement les valeurs du tableau, elle contient une indirection, une référence, une adresse vers le tableau.

notes

résumé

2m 1s

On suppose que l'adresse du tableau est ici. Et donc dans la variable « score » je stocke uniquement la référence vers ce tableau. En terme de jargon, on va dire que la variable « score » référence ou pointe vers un tableau de int référence ou pointe vers un tableau de int

notes

résumé

2m 49s