Atelier Programmation Orientée Objet Avancée –JAVA-

TP3 - Généralité

LES CHAÎNES DE CARACTÈRES

Nous aborderons dans cet atelier les objets String & Autres

ISET Zaghouan

Enseignant

Boukchim_mossaab@yahoo.fr

La théorie

I. La définition d'une chaîne

Exemple :

String texte = "bonjour";

Les variables de type String sont des objets. Partout ou des constantes chaînes figurent entre guillemets, le compilateur Java génère un objet de type String avec le contenu spécifié. Il est donc possible d'écrire : **String texte = "Java Java Java".replace('a','o');**

Les chaînes ne sont pas des tableaux : il faut utiliser les méthodes de la classe String d'un objet instancié pour effectuer des manipulations.

Il est impossible de modifier le contenu d'un objet String construit à partir d'une constante.

Cependant, il est possible d'utiliser les méthodes qui renvoient une chaîne pour modifier le contenu de la chaîne

Exemple :

```
String texte = "Java Java Java";
texte = texte.replace('a','o');
```

Java ne fonctionne pas avec le jeu de caractères ASCII ou ANSI, mais avec Unicode (Universal Code). Ceci concerne les types char et les chaînes de caractères. Le jeu de caractères Unicode code un caractère sur 2 octets. Les caractères 0 à 255 correspondent exactement au jeu de caractères ASCII.

II. L'addition de chaînes

Java admet l'opérateur + comme opérateur de concaténation de chaînes de caractères.

L'opérateur + permet de concaténer plusieurs chaînes. Il est possible d'utiliser l'opérateur +=

Exemple :

```
String texte = "";
texte += "Hello";
texte += " World3";
```

Cet opérateur sert aussi à concaténer des chaînes avec tous les types de bases. La variable ou constante est alors convertie en chaîne et ajoutée à la précédente. La condition préalable est d'avoir au moins une chaîne dans l'expression sinon le signe "+" est évalué comme opérateur mathématique.

Exemple :

```
System.out.println("La valeur de Pi est : " + Math.PI);
int duree = 121;
System.out.println("durée = " + duree);
```

III. La comparaison de deux chaînes

Il faut utiliser la méthode **equals()**

Exemple :
```
String texte1 = "texte 1";
String texte2 = "texte 2";
if ( texte1.equals(texte2) )...
```

IV. La détermination de la longueur d'une chaîne

La méthode **length()** permet de déterminer la longueur d'une chaîne.

Exemple :
```
String texte = "texte";
int longueur = texte.length();
```

V. La modification de la casse d'une chaîne

Les méthodes Java **toUpperCase()** et **toLowerCase()** permettent respectivement d'obtenir une chaîne tout en majuscule ou tout en minuscule.

Exemple :
```
String texte = "texte";
String textemaj = texte.toUpperCase();
```

VI. Caractère et sous-chaîne

1. char charAt(int i)

Retourne le caractère à l'indice spécifié en paramètre.

2. String substring(int d)

Sous-chaîne depuis d jusqu'à la fin

3. String substring(int d, int f)

Sous-chaîne depuis d jusqu'au caractère d'indice f non inclus.

VII. Conversions

1. String trim()

Retourne une chaîne égale à la chaîne sans les espaces de début et de fin.

2. String replace(char ac, char nc)

Retourne une chaîne ou tous les ac ont été remplacé par des nc. S'il n'y a pas de remplacement, la chaîne elle-même est retournée.

VIII. Recherche

1. **int indexOf(int ch)**

Retourne l'indice de la première occurrence du caractère

2. **int indexOf(int ch, int fromIndex)**

Retourne l'indice de la première occurrence du caractère à partir de fromIndex

3. **int indexOf(String str)**

Retourne l'indice de la première occurrence de la chaîne

4. **int indexOf(String str, int fromIndex)**

Retourne l'indice de la première occurrence de la chaîne à partir de fromIndex

5. **int lastIndexOf(int ch)**

Retourne l'indice de la dernière occurrence du caractère

6. **int lastIndexOf(int ch, int fromIndex)**

Retourne l'indice de la dernière occurrence du caractère à partir de fromIndex

7. **int lastIndexOf(String str)**

Retourne l'indice de la dernière occurrence de la chaîne

8. **int lastIndexOf(String str, int fromIndex)**

Retourne l'indice de la dernière occurrence de la chaîne à partir de fromIndex

Les exercices

Exercice 1 :

Il s'agit d'écrire un programme qui, étant donnée une chaîne de caractères (une instance de la classe String)

- calcule la chaîne inverse
- indique s'il s'agit ou non d'un palindrome

Exercice 2 :

Ecrire une méthode qui à une chaîne de caractères associe le nombre de mots que celle-ci contient. Ecrire ensuite une méthode qui calcule la longueur moyenne des mots d'un texte.

Exercice 3 :

Ecrire un programme qui lit une chaîne de caractères et affiche le pourcentage des lettres voyelles.

Exercice 4 :

Ecrire un programme qui lit une chaîne de caractères contenant des parenthèses ouvrantes et fermantes et doit affiché si le parenthèsage est cohérent ou non.

Devoir à domicile

Exercice 1 :

Ecrire une méthode qui à une chaîne de caractères associe le nombre de caractères distincts que celle-ci contient.

Exercice 2 :

Ecrire un programme qui demande à l'utilisateur un texte et un mot, puis affiche pour chaque lettre du mot, le nombre d'occurrences de cette lettre dans le texte de départ.

Exercice 3 :

Ecrire une méthode qui à deux chaînes de caractères s1 et s2 associe la chaîne constituée des caractères de s1 qui n'apparaissent pas dans s2 (dans le même ordre).
fermantes et doit affiché si le parenthèsage est cohérent ou non.

Annexe « La classe Scanner »

Elle simplifie la lecture de données sur l'entrée standard (clavier) ou dans un fichier.

Pour utiliser la classe Scanner, il faut d'abord l'importer :

- ✓ import java.util.Scanner;

Ensuite il faut créer un objet de la classe Scanner :

- ✓ Scanner sc = new Scanner(System.in);

Pour récupérer les données, il faut faire appel sur l'objet sc aux méthodes décrites ci-dessous.

Ces méthodes parcourent la donnée suivante lue sur l'entrée et la retourne :

- ✓ String next() donnée de la classe String qui forme un mot,
- ✓ String nextLine() donnée de la classe String qui forme une ligne,
- ✓ boolean nextBoolean() donnée booléenne,
- ✓ int nextInt() donnée entière de type int,
- ✓ double nextDouble() donnée réelle de type double.

Il peut être utile de vérifier le type d'une donnée avant de la lire :

- ✓ boolean hasNext() renvoie true s'il y a une donnée à lire
- ✓ boolean hasNextLine() renvoie true s'il y a une ligne à lire
- ✓ boolean hasNextBoolean() renvoie true s'il y a un booléen à lire
- ✓ boolean hasNextInt() renvoie true s'il y a un entier à lire
- ✓ boolean hasNextDouble() renvoie true s'il y a un double à lire.