

Manipulations de Strings

```
String inputString = "ceci est mon string, 32";
String[] tab = inputString.split(", ");
String str = tab[0].toUpperCase();
int number = Integer.parseInt(tab[1]);
System.out.printf("%s = %d %n", str, number);

// Tester des strings
boolean finiParTest = inputString.endsWith("test"); // false
boolean commenceParCeci = inputString.startsWith("ceci"); // true

// Avoir la longueur d'un string
System.out.println(inputString.length()); // 23

// Utiliser substring pour diviser un string
String hhmm = "10:30";
System.out.println(hhmm.substring(0, hhmm.length() - 3)); // 10
System.out.println(hhmm.substring(hhmm.length() - 2, hhmm.length())); // 30

// Trouver la position d'un caractère dans un String
int position = hhmm.indexOf(':'); // 2

// Retrouver le caractère à un certain indice de position
char caractere = hhmm.charAt(position); // ':'
System.out.printf("Le caractère à %d est '%c'", position, caractere); // "Le caractère en position 2 est ':'"

// Transformer un int en String
int nombre = 42;
String nombreEnString = String.valueOf(nombre); // "42"
```

String étant une classe, elle contient plusieurs méthodes, en voici 2 qui soit très utiles :

- `.split("regex")` pour diviser un String en un tableau en utilisant un autre String comme délimiteur de la colonne.
- `.toUpperCase()` et `.toLowerCase()` pour convertir un String en majuscule ou minuscule

- `.endsWith()` et `.startsWith()` pour tester si un String termine ou commence par un certain substring.
- `.length()` pour obtenir la longueur de la chaîne de caractères.
- `.substring()` pour diviser un string avec 2 positions. Cette méthode prends 2 arguments, la première position (qui est inclusive, donc prise en compte dans le résultat), et la deuxième position qui est exclusive (donc non prise en compte dans le résultat). Donc si on a 0 en premier argument et 5 en deuxième argument, on aura les caractères 1, 2, 3 et 4.
- `.indexOf()` permet de savoir quel est l'indice de position d'un caractère ou un String dans un String.
- `.charAt()` est l'inverse de `indexOf` et retourne un caractère à une position donnée
- `.valueOf()` permet de transformer un autre type en String.

“ A noter que l'utilisation des méthodes String indiquées ici ne transforme pas la variable String d'origine, elle ne fait que retourner une nouvelle valeur.

En savoir plus

- [Oracle Docs - String](#)

Revision #1

Created 27 April 2023 04:31:26 by SnowCode

Updated 27 April 2023 04:35:12 by SnowCode