Strings

Mise à niveau Java

Y. Boichut & F. Moal

Strings

- 2 types de strings
  - *String* pour les chaînes constantes
  - *StringBuilder* ou *StringBuffer* pour les chaînes variables

L’objet String

- Affectation
  - chaine = “ma string”;
- En Java, les deux variables ci-dessous vont référencer le même objet
  - chaine1 = “maString”;
  - chaine2 = “maString”;

Ainsi, *chaine1 == chaine2*

- Par contre ça n’aurait pas été le cas si nous avions eu : *chaine2 = new String(“maString”)*;

Concaténation

- opérateur de concaténation ‘+’

  int x=5;

  s = “Valeur de x” + x;

  - Types primitifs sont gérés par le compilateur
  - pour les instances de classes autres, la méthode toString() est automatique appelée

Egalité de Strings

- La méthode *equals* teste si deux instances de String ont sémantiquement la même valeur
  - (“salut “+”la compagnie”).equals(“salut la compagnie”) => TRUE
  - String c1 = “salut ”;
  - (c1+”la compagnie”).equals(“salut la compagnie”) => TRUE
  - (c1+”la compagnie”) == “salut la compagnie” => FALSE
- *equalsIgnoreCase()* est un test d’égalité en ne tenant pas compte des minuscules et des majuscules

Comparaison lexicographique

- c1.compareTo(c2) renvoie le signe de “c1-c2”
  - 0 : égalité entre c1 et c2
  - <0 : c2 suit c1 dans l’ordre lexicographique (c1 < c2)
  - >0 : c2 précède c1 dans l’ordre lexicographique (c1 > c2)
- c1.compareToIgnoreCase(c2) fait la même chose en ignorant les majuscules et minuscules

Quelques méthodes/fonctions de la classe String

- int indexOf(String s), int indexOf(String s, int d)
- int lastIndexOf(String s), int lastIndexOf(String s, int d)
- String substring(int d, int f), String substring(int d)
- boolean startsWith(String s), boolean startWith(String s, int i), boolean endsWith(String s)
- String trim()
- String toUpperCase()
- String toLowerCase()
- String valueOf(<Type primitif> t) - méthode statique

Exercice

Soit une classe Url qui possède deux champs - String protocole et String[] adresse. Le tableau contient chaque composant de l'adresse. Par exemple, pour l'adresse http://www.je-comprends.org/que/dalle, le tableau contiendra dans la case 0 : "www.je-comprends.org", dans la case 1 : "que"... Le champ protocole sera alors initialisé avec la valeur "http".

- Ecrire la classe et ses constructeurs de bases
- Ecrire la méthode void initialiseURL(String url) qui permettra d'initialiser à partir d'une URL (que nous supposerons correcte) de remplir les champs spécifiques

Strings modifiables

- *StringBuffer / StringBuilder* : le dernier a été ajouté dans la JDK 5 et reste stable dans un environnement concurrent (contrairement au premier)
- *append*
- *insert*
- *replace*
- *charAt*
- *substring*
- *reverse*
- Pas de *equals* définit dans ces classes