Informatique 1
L1 Portail IE
L1 Portail MA

Responsables:
pierre-alain.fouque@univ-rennes1.fr
patrick.derbez@univ-rennes1.fr

Plusieurs slides de F. Morain

Les chaînes de caractères

Le type **`char`** : les constantes s'écrivent 'a', 'A', '0', '+', ... Il existe des caractères spéciaux :

`'''`, `'\n'`, `'\r'`, `'\t'`, `'\'`.

- Unicode: codage des caractères sur 16 bits (vs. 8 bits en ASCII) qui permet le codage de tous les types d'alphabet.
- Les caractères utiles pour le français sont de code compris entre `\u0020` (pour espace) et `\u00FC` (pour ü), le tout en hexadécimal.

Les chaînes de caractères : **`String`** est une classe, ce n'est pas un type primitif.

Manipulation

Comment créer une **`String`**:

- **`String s = "Bonjour!";`**
- par concaténation: **`s = "Bonjour" + "!";`**
- par conversion: **`s = String.valueOf(i);`**
- **`...main(String[] args)`** : **`args`** contient les arguments passés sur la ligne de commande. Par exemple:

```
public static void main(String[] args){
  System.out.println(args[0]+" "+args[1]);
}
```

affiche les deux premiers arguments. De même:

```
int n = Integer.parseInt(args[0]);
String s = args[1];
```

permet de récupérer un entier passé en argument sur la ligne de commande Unix:

```
java Essai 10 oui
```

Méthodes

- **`s.length()`** retourne la longueur de **`s`** (au lieu de **`t.length`** pour un tableau) ;
- **`s.charAt(i)`** retourne le i-ième caractère de **`s`** (compté à partir de 0) ;
- **`s.equals(t)`** : retourne `true` si **`s`** et **`t`** contiennent les mêmes caractères; plus précis **`s.compareTo(t)`** ;
- Immutable: on ne peut pas écrire

```
String s = "Bonjour!";
s.charAt(0) = 'S';
```

```
Bug.java:4: unexpected type
required: variable
found   : value
        s.charAt(0) = 'S';
                    ^
1 error
```

String ou char[]

```
char[] t = {'B', 'o', 'n', 'j', 'o', 'u', 'r'};
String s = "Bonjour";

//Affiche la première lettre de t
System.out.println("La première lettre de t est: " + t[0]);

//Affiche la première lettre de s
System.out.println("La première lettre de s est: " + s.charAt(0));

//Affiche t
for (int i = 0; i < t.length; ++i) System.out.print(t[i]);
System.out.println();

//Affiche s
System.out.println(s);
```

Affectation

```
String s1 = "Bonjour";
String s11 = "Bonjour";
String s2 = new String("Bonjour");
String s22 = new String("Bonjour");

System.out.println(s1);
System.out.println(s11);
System.out.println(s2);
System.out.println(s22);
```

```
Bonjour
Bonjour
Bonjour
Bonjour
```

Attention

```
String s1 = "Bonjour";
String s11 = "Bonjour";
String s2 = new String("Bonjour");
String s22 = new String("Bonjour");

System.out.println(s1 == s11);
System.out.println(s1 == s2);
System.out.println(s2 == s22);
```

Attention

```
String s1 = "Bonjour";
String s11 = "Bonjour";
String s2 = new String("Bonjour");
String s22 = new String("Bonjour");

System.out.println(s1 == s11);
System.out.println(s1 == s2);
System.out.println(s2 == s22);
```

```
true
false
false
```

Test d'égalité

```
String s1 = "Bonjour";
String s2 = new String("Bonjour");

System.out.println(s1.equals(s2));

String s3 = "bonjour";

System.out.println(s1.equalsIgnoreCase(s3));
```

```
true
true
```

Comparaison des caractères

- Il est possible d'utiliser les opérateurs de comparaison sur les variables de type char.
- Les résultats sont basés sur le jeu de caractères Unicode

```
System.out.println('a' < 'b');
System.out.println('Z' < 'a');
System.out.println('+' < 'J');
```

```
true
true
true
```

Application

- Afficher les 26 lettres de l'alphabet

```
for (char lettre = 'a'; lettre <= 'z'; ++lettre) {
    System.out.print(lettre);
}
```

```
abcdefghijklmnopqrstuvwxyz
```

Est un chiffre

- Écrire une fonction qui prend un caractère en entrée et retourne vrai ssi ce caractère est un chiffre:

```
public static boolean estUnChiffre(char c) {
    return ('0' <= c) && (c <= '9');
}


public static void main(String[] Args) {

    System.out.println(estUnChiffre('a'));
    System.out.println(estUnChiffre('5'));
}
```

```
false
true
```

Conversion en chiffre

- Écrire une fonction qui prend un caractère c en entrée et qui retourne –1 si c n'est pas un chiffre ou l'entier correspondant sinon:

```
public static int toInt(char c) {
    int res = -1;
    if ('0' <= c && c <= '9') res = c - '0';
    return res;
}


public static void main(String[] Args) {

    System.out.println(toInt('a'));
    System.out.println(toInt('5'));
}
```

char

- Afficher toutes les valeurs possibles d'une variable de type char:

```
char lettre = 0;
do {
    System.out.print(lettre);
    lettre += 1;
} while (lettre != 0);
```

Comparaison des Strings

- Attention, on ne peut pas utiliser les opérateurs de comparaison pour comparer des Strings:

```
System.out.println("a" < "b");
```

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        The operator < is undefined for the argument type(s) java.lang.String, java.lang.String
```

Comparaison des Strings

- On utilise l'ordre lexicographique:

```
public static int compare(String s1, String s2) {
    //retourne x tel que:
    // x < 0  <==> s1 < s2
    // x == 0  <==> s1 = s2
    // x > 0  <==> s1 > s2

    int min_length;
    if (s1.length() < s2.length()) min_length = s1.length();
    else min_length = s2.length();

    int i = 0;
    while (i < min_length && s1.charAt(i) == s2.charAt(i)) i += 1;

    int res;

    if (i < min_length) { // donc les caractères diffèrent à la position i
        if (s1.charAt(i) < s2.charAt(i)) res = -1;
        else res = 1;
    }
    else { // une des chaînes est un préfixe de l'autre
        // ex: bon et bonjour
        if (s1.length() < s2.length()) res = -1;
        else if (s1.length() == s2.length()) res = 0;
        else res = 1;
    }

    return res;
}
```

Comparaison des Strings

```
public static int compare(String s1, String s2) {
    //retourne x tel que:
    // x < 0  <==> s1 < s2
    // x == 0  <==> s1 = s2
    // x > 0  <==> s1 > s2

    int min_length = s1.length() < s2.length() ? s1.length() : s2.length();
    for (int i = 0; i < min_length; ++i) {
        int x = s1.charAt(i) - s2.charAt(i);
        if (x != 0) return x;
    }
    return s1.length() - s2.length();
}


public static void main(String[] Args) {

    System.out.println(compare("Bonjour", "aaaaaa"));
    System.out.println(compare("Bonjour", "Bonjour"));
    System.out.println(compare("bonjour", "aaaaaa"));
}
```

```
-31
0
1
```

Comparaison des Strings

- Java propose des méthodes pour comparer deux Strings:

```
String s1 = "Bonjour";
String s2 = "tout le monde!";

System.out.println(s1.compareTo(s2));
System.out.println(s1.compareToIgnoreCase(s2));
```