Objets/Instances : projet *SetPoint*

Introduction

Dans le chapitre sur les variables, nous avions définit les différentes sortes de chocolats dans une boîte. Nous avions utilisé par commodité les notations première et seconde boîte, mais l'ordinateur, lui, ne sait pas que les chocolats sont contenus dans une boîte, et encore moins qu'il y a 2 boîtes différentes ! En terme informatique, ces boîtes seraient des *objets*, également appelés *instances*. A ce niveau vous pourriez confondre un objet avec une matrice… conteneurs de chocolats…mais les matrices sont des sortes d'objets. On parle de langage de programmation *orienté objet* pour Java, ce qui signifie qu'il travaille avec des objets, leur assignant des propriétés (par exemple contenir des chocolats au lait et noirs) et des comportements (se re-remplir automatiquement de chocolats une fois vide…). La pratique vous aidera à éclaircir de concept flou.

Déclaration (ou création de nouveaux objets)

- String teamName = new String() ;
- Dinosaure Trex = new Dinosaure();
- BoiteChocos auLait = new BoiteChocos();

Si vous ajoutez ces lignes dans un de vos programmes, il refusera de se compiler. Lui ne sait pas ce qu'est un Dinosaure() ou un BoiteChocos(). Pouvez-vous ne décrire un *szertysprzchut* de la planète *sakasksproucht* ? Non. Pouvez-vous me décrire … un chien … oui. De même pour l'ordinateur qui connaît la classe String car ses propriétés et son comportement ont été préalablement définis. (Dans l'API de Java).
Il est définit dans les propriétés de String qu'il reçoit comme argument (ce qui vient entre les parenthèses) un chaîne. Si vous essayez de lui assigner autre chose qu'une chaîne comme argument, il génèrera une erreur et le programme ne se compilera pas.
De même, par exemple, pour Point qui admet comme arguments ses coordonnées en abscisse et ordonnée. (Dans (O, i, j), O situé en haut à gauche de l'écran et les coordonnées des vecteurs i et j exprimées en pixels).

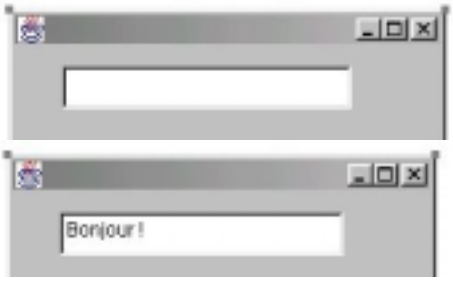
- Point A = new Point(10,3) ; // fonctionne
- Point B = new Point(" Bonjour ") ; // ne fonctionne pas

```
C:\WINDOWS\Bureau\JAVA\SITE\Cours\objets>javac Incompilable.java
Incompilable.java:6: Incompatible type for constructor. Can't convert java.lang.
String to java.awt.Point.
        Point B = new Point("Bonjour");
                            ^
1 error
```

Le compilateur nous dit qu'il ne peut pas convertir une valeur de type chaîne (String) en argument pour Point.

Accès aux valeurs

On peut par l'intermédiaire de méthodes (fonctions, ou série d'actions qui définissent un comportement) avoir accès aux valeurs des objets. On prendra pour exemple le composant TextField de l'interface utilisateur AWT de Java et lui assigner une valeur.



Le nom de cet objet est *textFiels1*. Il a été déclaré comme ceci :
TextField textField1 = new TextField();
On peut définir le texte qu'il affichera en utilisant la méthode *setText()* :
TextField1.setText("Bonjour !") ;
On a ainsi eu accès à sa valeur d'affichage de texte qui était auparavant fixée nulle.

Comparaison d'objets

On utilise les mêmes opérateurs que pour les variables, par exemple == pou égal ou != pour différent.

- Point A = new Point(10,3) ;
- Point B = new Point(11,4);
- System.out.println("Meme objet ?" + (A==B));

On compare ainsi les objets : « est-ce que A est B ? » et non pas « est-ce que A a la même valeur que B ? ».
Pour savoir s'ils ont la même valeur, on utilise la méthode *equals()*.

Application : le programme EgaliteTest

Dans ce programme nous allons revenir sur la comparaison des objets et la comparaison de leur valeur.
Note : souvenez vous dans le chapitre sur les variables : String n'en était pas une…

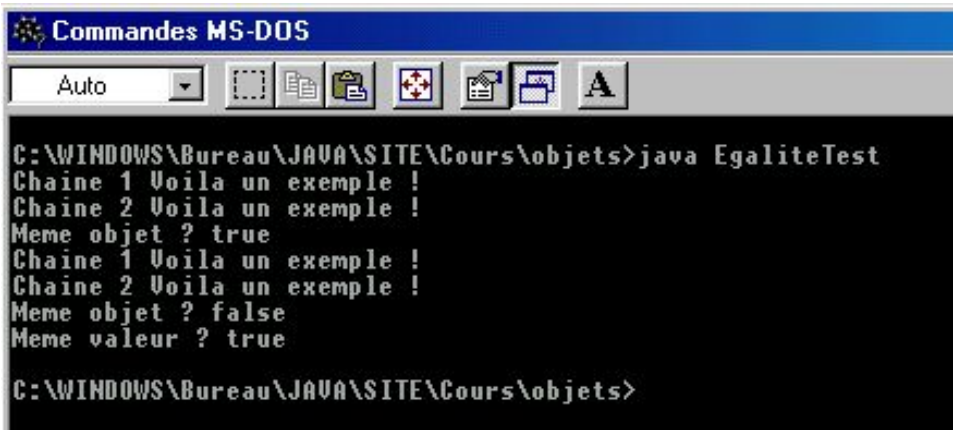
```
class EgaliteTest {
  public static void main(String args[]){
    String str1, str2; // 2 objets de type String
    str1 = "Voila un exemple !"; // valeur du premier

    str2 = str1; // le second est le premier

    System.out.println("Chaine 1 " + str1);
    System.out.println("Chaine 2 " + str2);
    System.out.println("Meme objet ? " + (str1 == str2));

    // on re-crée le second objet
    str2 = new String(str1); // le second a la valeur du premier

    System.out.println("Chaine 1 " + str1);
    System.out.println("Chaine 2 " + str2);
    System.out.println("Meme objet ? " + (str1 == str2));
    System.out.println("Meme valeur ? " + str1.equals(str2));
  }
}
```



La partie concernant les objets est énorme : c'est normal, java est un langage orienté objet… Nous verrons de nouvelles applications dans d'autres chapitres.