

Leçon N°3

Les énumérations

Introduction

Les énumérations sont des structures qui définissent une liste de valeurs possibles. Cela nous permet de créer des types de données personnalisés. Nous pouvons par exemple construire un type `Langage` qui ne peut prendre qu'un certain nombre de valeurs : `JAVA`, `PHP`, `C`, etc.

Déclaration

Une énumération se déclare comme une classe, mais en remplaçant le mot-clé `class` par `enum`. Autre différence : les énumérations héritent de la classe `java.lang.Enum`.

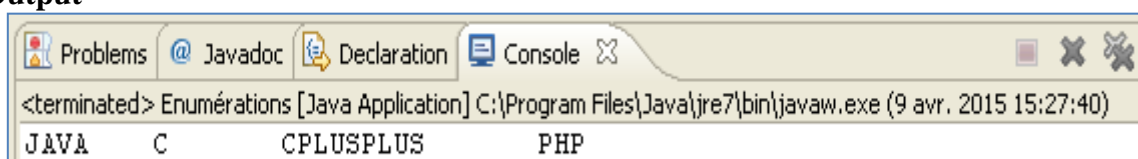
Voici à quoi ressemble la déclaration d'une énumération :

```
public enum Langage {
    JAVA,
    C,
    CPLUSPLUS,
    PHP ;
}
```

Avec cela, nous obtenons une structure de données qui encapsule quatre « objets ». En fait, c'est comme si vous aviez un objet `JAVA`, un objet `C`, un objet `CPLUSPLUS` et un objet `PHP` partageant tous les mêmes méthodes issues de la classe `java.lang.Object` comme n'importe quel autre objet : `equals()`, `toString()`, etc. De plus, nous pouvons recourir à la méthode `values()` qui retourne un tableau contenant la liste des déclarations de l'énumération comme dans l'exemple suivant :

```
public class Enumérations {
    public enum Langage {JAVA, C, CPLUSPLUS, PHP ;}
    public static void main(String args[]){
        for(Langage lang : Langage.values())
            System.out.print(lang + "\t");
    }
}
```

Output



Remarques

- Les objets d'une énumération sont des données constantes ; de ce fait, elles sont notées en Majuscule.
- Une énumération ne peut pas être locale (c'est-à-dire déclarée dans une méthode). Au contraire, il est possible de déclarer une énumération dans un fichier séparé, comme une classe.

Exemple complet

Les énumérations sont en fait compilées sous forme de classes, éventuellement *internes*. Une énumération peut donc avoir des attributs, des constructeurs privés et des méthodes. Ses constructeurs sont obligatoirement privés car aucune nouvelle instance ne peut être créée.

Exemple

```
public enum Langage {
    //Objets directement construits
    JAVA("Langage JAVA", "Eclipse"),
    C ("Langage C", "CodeLite"),
    Cplusplus ("Langage C++", "Visual Studio"),
    PHP ("Langage PHP", "PSPad"); //le «;» met fin à la liste des constantes

    // Attributs
    private String nom = "";
    private String editeur = "";

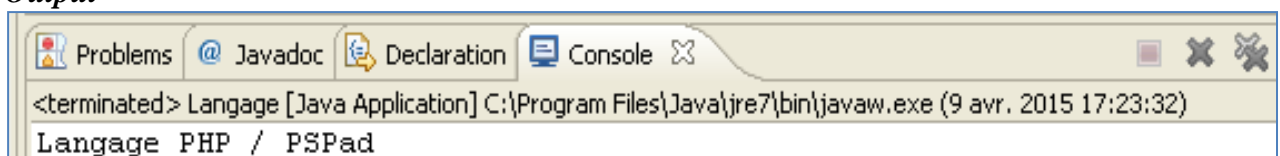
    //Constructeur
    private Langage(String nom, String editeur){
        this.nom = nom;
        this.editeur = editeur;
    }

    public String getEditeur(){
        return editeur;
    }

    public String toString(){
        return nom;
    }

    public static void main(String args[]){
        Langage l = Langage.PHP;
        System.out.println(l.toString()+" / "+l.getEditeur());
    }
}
```

Output



Méthodes utiles

Les énumérations possèdent toutes les méthodes suivantes :

Méthode	Rôle
<code>ordinal()</code>	Obtenir l'index de la valeur selon l'ordre de déclaration (premier = 0).
<code>name()</code>	Obtenir le nom de la valeur.
<code>valueOf(String s)</code>	Obtenir la valeur dont le nom est spécifié en paramètre (méthode statique).
<code>values()</code>	Obtenir un tableau contenant toutes les valeurs déclarées (méthode statique)
<code>compareTo</code>	Compare 2 objets d'une énumération sur la valeur de l'ordinal.

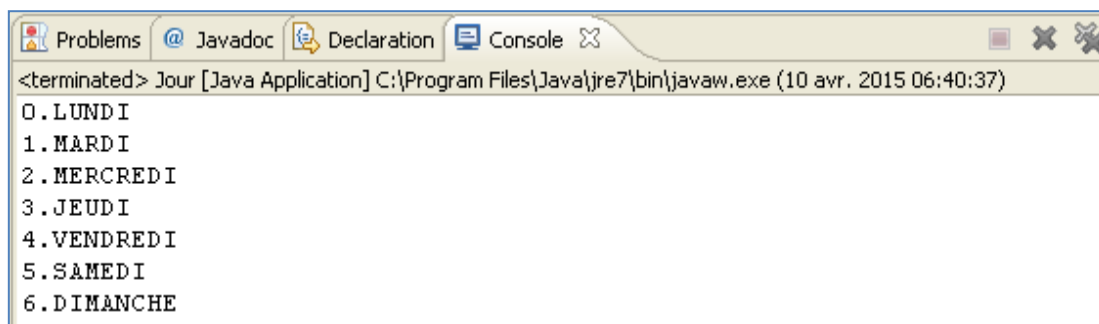
Les énumérations implémentent l'interface `java.lang.Comparable` et possèdent donc les méthodes `equals` et `compareTo` pour comparer deux valeurs (ordonnées selon `ordinal()` par défaut).

Exemple

```
public enum Jour {
    LUNDI,
    MARDI,
    MERCREDI,
    JEUDI,
    VENDREDI,
    SAMEDI,
    DIMANCHE;

    public static void main(String[] args){
        Jour[] tab = Jour.values();
        for (int i = 0; i < tab.length; i++)
            System.out.println(tab[i].ordinal()+"."+tab[i].name());
    }
}
```

Output



```
<terminated> Jour [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (10 avr. 2015 06:40:37)
0. LUNDI
1. MARDI
2. MERCREDI
3. JEUDI
4. VENDREDI
5. SAMEDI
6. DIMANCHE
```

Exercice

- Créer une énumération « SituationFamiliare » qui contient 4 valeurs (« CELIBATAIRE », « MARIE », « DIVORCE », « VEUF »)

- Créer une énumération « Titre » qui contient 3 valeurs (« MADAME », « MADEMOISELLE », « MONSIEUR »). Chaque titre possède une abréviation (« Mme », « Mlle », « Mr »).
Ajouter à cette énumération les méthodes suivantes :
 - Titre(abréviation) // constructeur privé
 - getAbréviation()
 - toString() : retourne l'abréviation
- Créer une classe « Personne » qui contient 4 attributs : titre, nom, age, étatCivil (sa situation familiale) ainsi que les méthodes suivantes :
 - Personne(titre, nom, age, etatCivil)
 - toString()
- Créer une classe « TestPersonnes » réduite à une méthode main() qui permet de :
 - Créer une collection de type ArrayList appelée « listePers » et y ranger 3 personnes
 - Parcourir la liste et afficher une description complète des personnes qui y figurent.