Datos de tipo enumerado en Java

Los tipos de datos enumerados son un tipo de dato definido por el programador (no como ocurre con los tipos de datos primitivos). En su definición el programador debe indicar un conjunto de valores finitos sobre los cuales las variables de tipo enumeración deberán tomar valores. La principal funcionalidad de los tipos de datos enumerados es incrementar la legibilidad del programa.

Este tipo de datos ha sido incorporado a Java a partir de la versión 1.5. y consiste en la posibilidad de definir nuevos tipos de datos cuyos posibles valores están limitados a un determinado conjunto dado.

Es una lista ordenada de elementos como constantes. A menos que se indique lo contrario, el primer miembro de un conjunto enumerado de valores toma el valor de 0 (definición general). Entre las llaves se ponen los posibles valores que podrán tomar las variables de tipo enumeración, valores que habitualmente se escriben en letras mayúsculas. Un ejemplo de enumeración podría ser:

```
public enum Semana {LUNES, MARTES, MIÉRCOLES, JUEVES,VIERNES, SÁBADO, DOMINGO}
```

Una enumeración es un tipo de datos, sus miembros son constantes que están escritas como identificadores, pero que toman valores enteros con signo.

El valor de una enumeración **no** puede ser leído desde el teclado y asignada a una variable de enumeración.

Las enumeraciones proporcionan a Java tipos enumerados, que definen clases representando un único elemento enumerado. No proporcionan ningún constructor público, solo métodos estáticos y finales.

Las definiciones de los tipos enumerados deben realizarse fuera del método main y, en general, fuera de cualquier método; es decir, deben realizarse directamente dentro del cuerpo de la clase.

Se utiliza la palabra reservada **enum** para definir datos de este tipo, por ejemplo:

```
enum Orientacion {NORTE, SUR, ESTE, OESTE}
```

Para versiones anteriores a la de J2SE 1.5, la forma de definir un determinado conjunto de valores era mediante la utilización de variables finales (constantes en otros lenguajes):

```
public class Orientacion
{ public static final int NORTE=1;
  public static final int SUR=2;
  public static final int ESTE=3;
  public static final int OESTE=4;
}
```

Para referirse a alguno de los valores definidos en las constantes anteriores se utiliza la sintaxis:

```
Nombre_clase.Constante;
```

Por ejemplo, Orientacion.ESTE hace referencia al valor 3.

Definición de un tipo enumerado

```
[public] enum Nombre_tipoEnumerado

{ VALOR1, VALOR2, .. , VALORn;

}
```

La declaración de un tipo enumerado puede estar en el interior de una clase o en el exterior de ésta, incluso puede estar en un archivo separado, pero nunca dentro de un método. Para el último caso, el nombre del archivo será el mismo que el del tipo enumerado y llevará la extensión .java, y se compilará como cualquier archivo .java.

Por ejemplo, para el caso que se presente al principio de este tema, el tipo enumerado quedará de la forma siguiente:

```
enum Orientacion

        {NORTE, SUR, ESTE, OESTE;}          //El ; (punto y coma) puede ser opcional.
```

Una enumeración es un tipo especial de clase, clase que hereda de java.lang.Enum. A diferencia de las clases estándares, una clase de enumeración no permite el uso del operador new para la creación de objetos. Cada uno de los valores de la enumeración representa uno de los posibles objetos de la clase, los cuales serán creados de forma implícita al hacer referencia en el código estos valores.

Además de los métodos heredados por la clase Enum, las enumeraciones disponen del método values( ), que devuelve un arreglo con todos los objetos de la clase.

Creación de un objeto enumeración

Una vez que se ha definido la enumeración, se puede crear una variable de ese tipo, aun que se trata de una clase, no se requiere del operador new como en las clases comunes, para este caso solo se requiere declarar una variable del tipo de la enumeración en cuestión y se le asigna el valor que se requiere, como si se tratara de un dato de tipo primitivo, por ejemplo:

```
Orientacion orienta;   /* aquí solo se declara la variable, ya más adelante se le puede asignar un valor como si se tratara de un  dato primitivo, con la diferencia que siempre hay que hacer referencia del tipo que lo contiene. */

    orienta=Orientacion.NORTE;          // se asigna una constante de enumeración a la variable antes declarada.

  Orientacion orienta=Orientacion.SUR;   //aquí se declara una variable y se le asigna un valor, es decir, se inicializa.
```

Para mostrar todos los valores definidos en una enumeración, siguiendo el ejemplo anterior tenemos:

```
        for(Orientacion orienta:Orientacion.values())

        {

                System.out.println(orienta);

        }
```

Como se ha visto, para darle un valor a las variables de tipo enumeración éstas deben asignarse a uno de los valores creados en su definición. El nombre del valor debe ir precedido del nombre de la propia enumeración:

```
        variable = Semana.DOMINGO;
```

Las enumeraciones a diferencia de los objetos, pueden compararse a través del operador = = o haciendo uso del método equals(), además de utilizarse como expresión en la instrucción switch, para este último, cuando se define la sentencia case, solo se pone el nombre de la constante de enumeración sin la referencia del tipo que la define, si se hace la referencia, marca error de sintaxis al momento de la compilación, es decir pondríamos **case SUR**, en lugar de **case Orientacion.SUR**.

```
switch(orienta)
{
        case NORTE : sentencias; break;
        case SUR : sentencias; break;
        :
}
```

Métodos

Existen métodos para el manejo de datos enumerados, entre estos se encuentran:

values() → devuelve todos los valores de la enumeración en una colección

valueOf(cadena) → devuelve la constante de enumeración que corresponde a un parámetro de tipo String.

Orientacion o=Orientacion.valueOf(”SUR”);

Regresa: Orientacion.SUR, la cual se almancena en la variable enumerada o

name() → devuelve el nombre de la constante en la enumeración.

Orientacion.SUR.name();

Regresa: SUR

ordinal() → devuelve la posición de la constante declarada, siendo la inicial 0.

Orientacion.SUR.ordinal();

Regresa: 1 ya que SUR es la segunda de la lista

compareTo() → Compara el objeto actual con el objeto de la enumeración que se especifique, devolviendo un número negativo, cero, o un número positivo dependiendo de que el objeto actual sea menor, igual o mayor que el objeto especificado de la enumeración, respectivamente.

Orientacion.SUR.compareTo(Orientacion.OESTE);

Regresa: -2, porque SUR está dos posiciones más atras que OESTE en la declaración de la enumeración.

toString() → regresa una cadena la cual correspondo con el objeto enumerado contenido en la variable enumerada.

Orientacion c=Orientacion.SUR;

String cad=c.toString();

Regresa ”SUR” el cual se almacena en la variable cad.

Resumen

En resumen, las características de los tipos enumerados en Java son:

- Son clases especiales de java y hereden implícitamente de la clase java.lang.Enum.
- No tienen constructores, aunque sean objetos, no tienen el método constructor asociado a la creación de objetos, pero si como parte de su diseño.
- Son public, static y final. Con el modificador de acceso public nos da las facilidades para que puedan ser accedidos sin restricciones. Al ser de tipo static, sus valores no pueden cambiar una vez definidos, y son de tipo final porque, aunque sean clases, no pueden ser heredadas.
- Tienen métodos asociados, como los que se acaban de mencionar en el apartado anterior.
- Al ser al final de cuentas una clase, un dato enumerado en Java puede contener la definición de variables de instancia y métodos, e incluso al método main().
- Es muy común hacer uso del ciclo for-each para recorrer los datos que componen a la enumeración.
- A cada elemento que se define en la enumeración se le asocia internamente un valor entero positivo, comenzando desde 0 (cero).
- A los elementos de una enumeración se les puede conocer como *constantes de enumeración*. Y cada uno está implícitamente declarado como public static final de la clase (enumeración) que lo contiene. Su tipo es el tipo de la enumeración que lo contiene, por ejemplo, la constante de enumeración SUR, pertenece al tipo Orientacion.
- No pueden heredar ni ser heredadas.

Otros Ejemplos

```
public enum Estaciones
 {PRIMAVERA, VERANO, OTOÑO, INVIERNO}

public enum DiasSemana

{LUNES, MARTES, MIÉRCOLES, JUEVES, VIERNES, SÁBADO, DOMINGO}


public class Ejemplo
{
	//definimos un tipo enumerado
	//los tipos enumerados deben definirse siempre fuera
	//del main y, más en general, fuera de cualquier método public

	 enum Semana {LUNES, MARTES, MIERCOLES, JUEVES,VIERNES, SABADO, DOMINGO};
	public static void main(String[] args)
	 {
		//definimos una variable que pertenece al tipo enumeradoSemana
		//y le damos el valor que representa el día martes
		Semana hoy = Semana.MARTES;//si el día se cayese en el fin de semana no hay que trabajar

		if (hoy == Semana.DOMINGO || hoy == Semana.SABADO)
		{
			System.out.println("Hoy toca descansar");
		}
		 else
		{
			System.out.println("Hoy toca trabajar");
		}
	}
}
```