M.3611.056.003 - Curso de Introducción a la programación con JAVA (Presencia Virtual Online - Cádiz)

POO2: Enumeradores, Paquetes, Modificadores

Eduardo Corral Muñoz
eoi_eduardo@corral.es

Programación Orientada a Objetos 2

Índice

— 01

Enumeradores

Enumeradores

- **enum** es un tipo especial de clase que representa un conjunto de constantes públicas.
- Crear un enum:

```
public enum Color{
    ROJO,
    VERDE,
    AZUL,
    AMARILLO,
    BLANCO,
    NEGRO
}
```

- El valor de cada constante es su propio nombre.

Enumeradores

- Para utilizarlos en una clase, hemos de declarar las variables utilizando como tipo en nombre del enumerador y para asignar el valor los haremos con el nombre del enumerador seguido del nombre de la constante:

  NombreEnum variable = NombreEnum.CONSTANTE;

- Pueden contener atributos y métodos como las clases, pero son **public**, **static** y **final**. (Inalterables).
- No pueden crear objetos y no pueden “extender” otras clases (pero si implementar interfaces).
- enum se utiliza cuando hay grupos de valores constantes como los días de la semana, los meses, …

Enumeradores

- En una clase:

```
Color miColor = Color.VERDE;
```

- Podemos recorrer los valores con un foreach

```
…
for(Color elColor: Color.values()){
      System.out.println(elColor);
}
…
```

Enumeradores

- En una estructura switch:

```
switch(miColor){
    case ROJO:
        …
        break;
    case VERDE:
        …
        break;
    ….
}
```

Enumeradores

Un enumerador más versátil, con más atributos y métodos

```
public enum TipoAutomovil {

    SEDAN("Sedan", “Mediano", 4),
    STATION_WAGON(“Familiar", “Grande", 5),
    HATCHBACK("Hatchback", “Compacto", 5),
    PICKUP("Pickup", "Camioneta", 4),
    COUPE("Coupé", “Pequeño", 2),
    CONVERTIBLE("Convertible", “Deportivo", 2),
    FURGON("Furgón", “Utilitario", 3),
    SUV("SUV", "Todo terreno deportivo", 5);

    private final String nombre;
    private final int numeroPuerta;
    private final String descripcion;

    …
}
```

Enumeradores

Un enumerador más versátil

```
public enum TipoAutomovil {
    …  // Constructor y getter
    TipoAutomovil(String nombre, String descripcion, int numeroPuerta) {
        this.nombre = nombre;
        this.numeroPuerta = numeroPuerta;
        this.descripcion = descripcion;
    }
    public String getNombre() {
        return nombre;
    }
    public int getNumeroPuerta() {
        return numeroPuerta;
    }
    public String getDescripcion() {
        return descripcion;
    }
}
```

Enumeradores

- Para utilizarlo dentro de una clase hemos de definir un atributo del tipo del enumerador:

private NombreEnum variable;

- Añadimos métodos getter y setter.
- En el main de nuestra clase principal asignamos el valor del atributo para cada objeto con su setter correspondiente:

objeto.setVariabe(NombreEnum.VALOR);

- Para obtener los atributos del enumerador:

objeto.getVariable().getAtributo();

— 02

Diagramas UML

Diagramas UML

- Nos permiten definir las clases que vamos a utilizar en un proyecto y crear diagramas de las relaciones entre ellas.
- Hay diversas herramientas disponibles, unas gratuitas y otras bajo licencia.

Umbrello – The UML Modeller

— 03

Paquetes

Paquetes

- Un paquete (package) es un contenedor en el que se agrupan y organizan clases relacionadas entre sí, como si las guardáramos en un misma carpeta.
- Se emplean para evitar conflictos en los nombres (mismo nombre de clase en diferentes paquetes, i.e. Date) y mantener mejor el código.
- Hay dos categorías de paquetes:

  Paquetes incorporados (**Built-in** packages) → Java API

  Paquetes definidos por el usuario (**User-defined**)

Paquetes

- El estándar para los nombres es que se escriban en minúsculas, tanto los paquetes como los subpaquetes.
- Los subpaquetes se separan con “.” (i.e. java.util)
- La empresas suelen nombrar sus paquetes empleando los nombres de sus dominios de Internet.

com.google

com.facekook

es.corral.eoi.proyecto1

es.corral.eoi.proyecto2

es.corral.eoi.proyecto3.compras

es.corral.eoi.proyecto3.ventas

Paquetes – Built-in packages

- El API de Java es una biblioteca de clases predefinidas incluidas en el JDE.
- Contiene componentes para manejar entradas, programar bases de datos, servidores web y muchísimas cosas más.
- Está dividida en packages y classes, y se pueden importar clase a clase o los paquetes completos.

```
import package.name.Class;

import package.name.*;
```

Paquetes – User packages

- Java emplea el sistema de archivos para almacenarlos (estructura de carpetas).

root → mipaquete → MiClase.java

- Para crear un paquete se emplea la palabra clave package seguido del nombre del paquete como primera línea de código.

package es.corral.eoi

- Después, definimos la clase:

public class MiClase{…}

- Para utilizar la clase en otra, la importamos

import es.corral.eoi.MiClase

- Para utilizar todas las clases del paquete en una clase:

import es.corral.eoi.*

Paquetes – User packages

- Para utilizar la clase en otra, la importamos

import es.corral.eoi.MiClase

o utilizamos el nombre completo de la clase

es.corral.eoi.MiClase objeto = new es.corral.eoi.MiClase();

- Para utilizar todas las clases del paquete en una clase:

import es.corral.eoi.*

- También podemos importar métodos y atributos estáticos

import es.corral.eoi.MiClase.miMetodo;

…

Tipo variable = miMetodo();

Paquetes – User packages

- Creando el primer paquete

Nuevo proyecto → (src = root) new package => crea subcarpeta

(subcarpeta) new Class o new package

- Se crea una estructura de directorios que contiene las clases tal como las hemos creado.

— 04

Modificadores

Modificadores de acceso

- Los modificadores de acceso son palabras reservadas que definen el tipo de acceso que puede tener una clase, un atributo de clase o un método de clase.
- **public** - público: accesible para todas las clases.
- **private** - privado: accesible sólo dentro de la clase declarada.
- **protected** - protegido: accesible sólo dentro de la clase y subclases declaradas.
- **default** - predeterminado: cuando no se especifique, será el predeterminado y estará accesible en el mismo paquete

Modificadores de acceso

Aplicables a Clases		
public	La clase es accesible por cualquier otra clase	Ejemplo
defecto	Accesible solo por clases del mismo paquete (no espec.)	Ejemplo

Aplicables a atributos, métodos y constructores		
public	Accesible por cualquier otra clase	Ejemplo
private	Accesible solo dentro de la clase que está declarado	Ejemplo
defecto	Accesible en el mismo paquete (no espec.)	Ejemplo
protected	Accesible en el mismo paquete y subclases	Ejemplo

Modificadores de no-acceso

Aplicables a Clases		
final	La clase no puede ser heredada por otra clase	Ejemplo
abstract	La clase no se puede usar para crear objetos	Ejemplo

Aplicables a atributos, métodos y constructores	
final	No pueden ser sobreescritos o modificados
static	Atributos y métodos pertenecen a la clase, no al objeto
abstract	Solo métodos en clase abstracta. El método carece de body
transient	Atributos y métodos no se incluyen al serializar el objeto
synchronized	Solo un hilo puede acceder al método a la vez
volatile	El valor se lee siempre de la memoria principal (sin caché)

05

Ejercicio

Ejercicio

- Sistema de facturación.

Se trata de generar una factura partiendo de los datos del cliente, añadiendo una línea (item) por cada producto que haya comprado, calculando el importe de cada línea (unidades * precio) y el total de la factura.

La entrada de datos la haremos por consola, al igual que la salida del resultado final.