

0101010
0100101
1101010

UD6.1- Tipus de dades enumerats

Programació – 1er DAW/DAM

0. CONTINGUTS

- INTRODUCCIÓ
- TIPUS ENUMERATS
- OPERACIONS
 - COMPARACIONS
 - EIXIDA PER PANTALLA

1. INTRODUCCIÓ

- A la vida real, encontrem **característiques** que sols poden tenir un **conjunt finit de valors**:
 - **Tipus de cotxe**: MONOVOLUMEN, ESPORTIU, CROSSOVER.
 - **Mesos de l'any**: GENER, FEBRER, MARÇ,
 - **Cares d'una moneda**: CARA o CREU
 - **Fitxes del 3 en ratlla**: FITXA_X o FITXA_Y



1. INTRODUCCIÓ

Si pensem en una classe **Cotxe** on necessitem d'un **atribut tipus...** que sols pot tenir els següents **valors**:

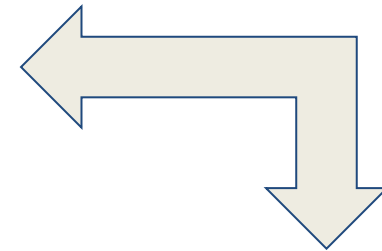
TURISME, MONOVOLUMEN, ESPORTIU, CROSSOVER

```
public class Coche {  
  
    private String tipoCotxe  
  
}
```

1. INTRODUCCIÓ

- Podem **declarar constants** i assignar a cada **tipus** un **enter** o un **String** que el representi

```
final String TURISMO = "turisme";  
final String MONOVOLUMEN = "monovolumen";  
final String DEPORTIU = "esportiu";  
final String CROSSOVER = "crossover";
```

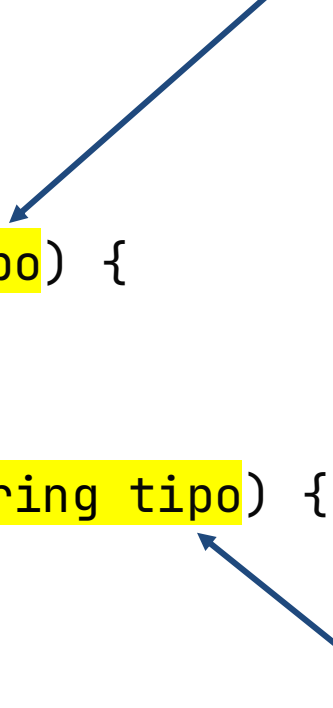


```
final int TURISMO = 1;  
final int MONOVOLUMEN = 2;  
final int ESPORTIU = 3;  
final int CROSSOVER = 4
```

1. INTRODUCCIÓ

- Encara que potser vàlid, presenta ~~certes limitacions~~.

```
public class Cotxe {  
    private String tipo;  
    public Cotxe(String tipo) {  
        this.tipo = tipo;  
    }  
    public void setTipo(String tipo) {  
        this.tipo = tipo;  
    }  
}
```



Com podem assegurar que l'atribut `tipo` sols continga els VALORS PERMESOS?



2. TIPUS ENUMERATS

- Els enumerats són **classes especials** que ens permeten definir un **nou tipus de dades** y els **possibles valors** que poden contenir.

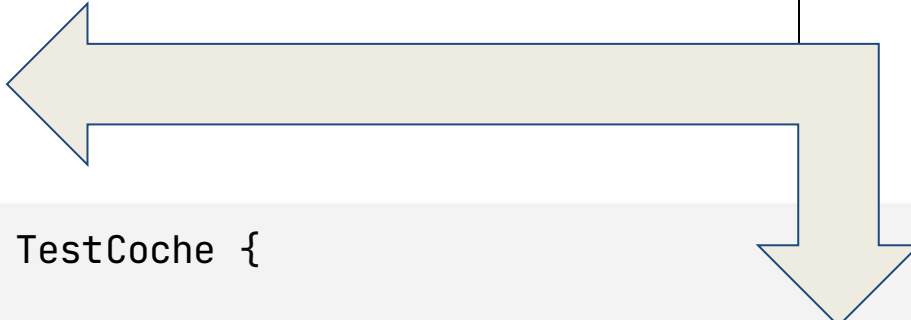
```
public enum Tipo {  
  
    TURISMO, MONOVOLUMEN, DEPORTIVO, CROSSOVER, TODO_TERRENO  
  
}
```

Al igual que les classes vistes fins ara, hem de declarar-los en un fitxer amb el mateix nom **Tipo.java**

2. TIPUS ENUMERATS

- Una volta declarat, disposarem d'aquest nou tipus de dades en el sistema

```
public class Cotxe {  
    private String marca;  
    // Variable del nuevo tipo creado  
    private Tipo tipo;  
  
    public Cotxe(Tipo tipo, String marca) {  
        this.tipo = tipo;  
        this.marca = marca;  
    }  
  
    public Cotxe(String marca) {  
        this.tipo = Tipo.TURISMO;  
        this.marca = marca;  
    }  
}
```



```
public class TestCoche {  
  
    public static void main(String[] args) {  
        Cotxe ibiza = new Cotxe(Tipo.DEPORTIVO, "Seat");  
        Cotxe picaso = new Cotxe(Tipo.MONOVOLUMEN,  
            "Citroen");  
    }  
}
```


2. OPERACIONS. Comparacions

- Podem dur a terme **comparacions** amb aquest tipus de dades

```
Cotxe ibiza = new Cotxe(Tipo.DEPORTIVO, "Seat");  
// Podemos llevar a cabo comparaciones en función del tipo  
switch (ibiza.getTipo()) {  
    case TURISMO:  
        System.out.print("Perfecto para el día a día");  
        break;  
    case TODO_TERRENO:  
        System.out.print("Perfecto para ir al campo");  
        break;  
    case MONOVOLUMEN:  
        System.out.print("Perfecto para viajar en familia");  
        break;  
    ...  
    default:  
        System.out.print("No existe el tipo");  
}
```

2. OPERACIONS. Eixida per consola I

- Si volem **convertir en text** un enumerat, **per defecte apareixerà** amb el text tal i com s'ha declarat.

```
public class TestCoche {  
  
    public static void main(String[] args) {  
        System.out.println(Tipo.TURISMO);  
    }  
}
```

Apareixerà **per pantalla** "TURISMO"

2. OPERACIONS. Eixida per consola II

- Podem canviar aquest comportament, fent ús del mètode màgic **toString** en el moment en el que declararem l'enumerat

```
public enum Tipo {  
    TURISMO {  
        @Override  
        public String toString() {  
            return "--turisme--";  
        }  
    },  
    MONOVOLUMEN {  
        @Override  
        public String toString() {  
            return "--monovolumen";  
        }  
    }  
    ...  
}
```

2. OPERACIONS. Eixida per consola III

- Sobre-escriptura del mètode **toString**

```
public class TestCoche {  
  
    public static void main(String[] args) {  
        System.out.println(Tipo.TURISMO);  
    }  
}
```

Apareixerà per pantalla "--turisme--"

3. Activitat Prèvia

Activitat 17. Defineix la classe **Cotxe** que apareix a les **diapositives anteriors**. Seguidament, afegeix un **nou atribut** a la classe Cotxe que indique si aquest és **automàtic** o **manual** (utilitza un enumerat).

Defineix els següents **constructors**:

- Constructor amb tots el atributs
- Constructor amb el tipus i la marca. Si el cotxe es un esportiu s'establirà per defecte el canvi automàtic, en cas contrari el canvi serà manual.
- Constructor amb la marca. S'establirà per defecte el tipus esportiu i canvi automàtic.

Per finalitzar, crea una classe **TestCotxe**, instancia **5 objectes** que facen ús dels constructors anteriors i mostra la informació de cada cotxe per pantalla

3. Activitat Prèvia

Això és tot ... de moment :-)