TIPOS ENUMERADOS

Un tipo enumerado en un tipo de dato definido por el usuario que solo puede tomar como posibles valores los definidos en una lista de constantes.

Un tipo enumerado se declara de forma general:

```
[modificadores] enum nombreEnum {VALOR1, VALOR2, VALOR3, ...}
```

modificadores (opcional) puede ser public, private, protected además de static.

enum es la palabra clave para definir enumeraciones en Java.

nombreEnum es el nombre del nuevo tipo creado.

VALOR1, VALOR2, ... son los valores que pueden tomar las variables que se declaren de este tipo.

Un enum se puede declarar dentro o fuera de una clase pero nunca dentro de un método.

Podemos considerar utilizar un tipo enumerado cuando una variable del programa solamente pueda tomar unos valores determinados.

Ejemplos:

```
enum DiaSemana {LUNES, MARTES, MIÉRCOLES, JUEVES, VIERNES, SABADO, DOMINGO};

enum Nivel {ALTO, MEDIO, BAJO};

enum Direccion {NORTE, SUR, ESTE, OESTE};
```

Por convenio se suelen escribir los valores del enum en mayúsculas ya que se trata de constantes.

Una vez creado el tipo enum ya podemos declarar variables.

Para declarar una variable de tipo DiaSemana escribimos:

```
DiaSemana d;
```

La variable d solo podrá tomar uno de los valores definidos en la lista de valores.

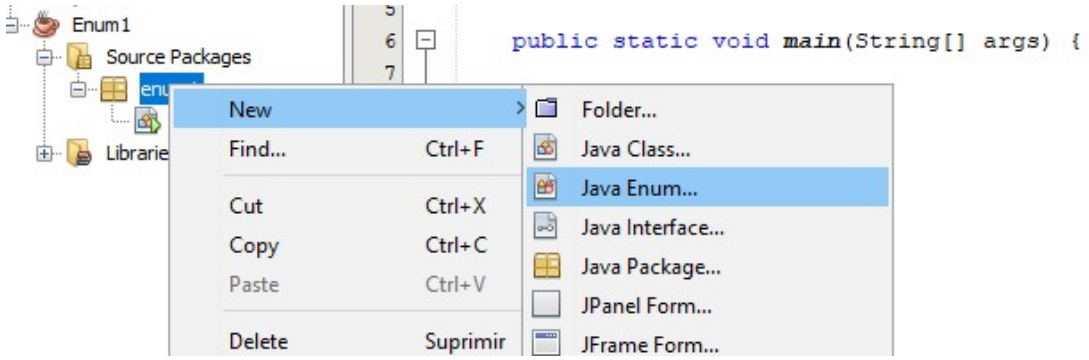
Para darle un valor a las variables lo haremos escribiendo *variable = nombreDelEnum.VALOR;*

Por ejemplo:

```
d = DiaSemana.JUEVES;
```

Ejemplo básico de uso de un tipo enumerado:

Creamos un enum llamado DiaSemana, si estamos haciendo un proyecto que contenga varias clases, lo normal es que el enum se escriba en un archivo aparte, de forma similar a como se hace con las clases.




```
public enum DiaSemana {
    LUNES, MARTES, MIERCOLES, JUEVES, VIERNES, SABADO, DOMINGO
}
```

Creamos una clase principal para utilizarlo:

```
public class EjemploEnum {
    public static void main(String[] args) {
        DiaSemana d; // declaramos una variable del tipo DiaSemana
        d = DiaSemana.DOMINGO; //asignamos a d un valor

        //Ejemplo de if para comparar una variable de tipo enum con un valor
        if (d == DiaSemana.DOMINGO || d == DiaSemana.SABADO){
            System.out.println("Estamos en fin de semana");
        }else{
            System.out.println("Aún no ha llegado el fin de semana");
        }
        //Ejemplo de uso de una variable de tipo enum en un switch
        switch (d) {
            case LUNES:
            case MARTES:
            case MIERCOLES:
            case JUEVES:
            case VIERNES:
                System.out.println("Aún no ha llegado el fin de semana");
                break;
            default:
                System.out.println("Estamos en fin de semana");
        }
    }
}
```

Los tipos enumerados en Java son mucho más potentes que sus equivalentes en otros lenguajes:

- **Un enum en Java es una Clase.**
- **Cada constante del enum es un objeto de la clase. NO se pueden crear nuevos objetos del enum. Todos los objetos de la clase son los que aparecen en la declaración del enum.**
- **Por ser una clase puede contener atributos y métodos**.
- **Todas las clases enum creadas heredan de forma implícita de la clase Enum de Java**

Algunos métodos que se heredan de Enum:

name() *public final String name()*

Devuelve un String con el valor de la constante que contiene tal y como aparece en la declaración del enum.

ordinal() *public final int ordinal()*

Devuelve un entero con la posición de la constante según está declarada en el enum. A la primera constante le corresponde la posición cero.

toString() *public String toString()*

Devuelve un String con el nombre de la constante que contiene tal y como aparece en la declaración del enum. Sobrescribe el método toString de la clase Object.

equals() *public final boolean equals(Object other)*

Devuelve true si los valores de dos enum son iguales. Este método sobrescribe el método equals de la clase Object. También se puede comprobar si dos enum son iguales **utilizando el operador ==**

compareTo() *public final int compareTo(Enum other)*

Compara el enum con el que recibe según el orden en el que están declaradas las constantes.

Devuelve un número negativo, cero o un número positivo según el objeto sea menor, igual o mayor que el que recibe como parámetro.

Solo se pueden comparar enumeraciones del mismo tipo.

valueOf() *public static enumConstant valueOf(String s)*

Devuelve la constante que coincide **exactamente** con el String que recibe como parámetro.

values() *public static enumConstant [] values()*

Devuelve un array que contiene todas las constantes de la enumeración en el orden en que se han declarado. Se suele usar en bucles for each para recorrer el enum.

http://docs.oracle.com/javase/8/docs/api/java/lang/Enum.html

Ejemplos de uso de estos métodos, utilizando el enum DiaSemana creado anteriormente:

```
public static void main(String[] args) {
        DiaSemana dia; //declaración de una variable tipo enum

        dia = DiaSemana.MIERCOLES; //Se le asigna un valor

        // dia.name() devuelve un String con el valor de la constante (MIERCOLES)
        System.out.println(dia.name());

        // día.toString() devuelve un String con el valor de la constante (MIERCOLES)
        System.out.println(dia.toString());

        // Devuelve un entero con la posición que ocupa dentro del enum (2).
        System.out.println(dia.ordinal());

        // Comprobar la igualdad mediante equals y ==
        if(dia.equals(DiaSemana.LUNES)){ //Esta condición no se cumple
            System.out.println("El valor es LUNES");
        }
        if(dia.equals(DiaSemana.MIERCOLES)){ //Esta condición se cumple
            System.out.println("El valor es MIERCOLES usando equals");
        }
        if(dia == DiaSemana.MIERCOLES){ //Esta condición se cumple
            System.out.println("El valor es MIERCOLES usando ==");
        }

        //Se crea otra variable y se le asigna un valor
        DiaSemana otroDia = DiaSemana.MARTES;

        // Comparamos las dos variables enum mediante compareTo
        // Se comparan según el orden en el que se han declarado en el enum
        // En este caso el if se cumple
        // dia contiene MIERCOLES y otroDia contiene MARTES
        // En la declaración del enum MIERCOLES aparece después de MARTES
        // por lo tanto es mayor
        if (dia.compareTo(otroDia) > 0) { //Esta condición se cumple: dia > otroDia
             System.out.println(dia + " > " + otroDia);
        } else {
             System.out.println(dia + " <= " + otroDia);
        }
        //Mostrar todos los elementos del enum
        //el método values() devuelve un array con todas las constantes del enum
        for (DiaSemana d : DiaSemana.values()) {
             System.out.print(d + " ");
        }
    }
```

Para **asignar valores a una variable enum desde teclado** podemos hacerlo de varias formas. Algunas de ellas aparecen en los siguientes ejemplos. Seguimos utilizando el enum DiaSemana.

Ejemplo 1: Leer un valor entero y hacer un switch. Según el valor introducido se asigna al enum la constante correspondiente.

```
Scanner sc = new Scanner(System.in);
int valor;
DiaSemana d = null; //Asigno null para evitar el warning de NetBeans: la variable d
                    //podría no estar inicializada
do {
    System.out.print("Introduce día de la semana del 1(Lunes) al 7(Domingo): ");
    valor = sc.nextInt();
} while (valor < 1 || valor > 7);

switch(valor){
       case 1: d = DiaSemana.LUNES;
               break;
       case 2: d = DiaSemana.MARTES;
               break;
       case 3: d = DiaSemana.MIERCOLES;
               break;
       case 4: d = DiaSemana.JUEVES;
               break;
       case 5: d = DiaSemana.VIERNES;
               break;
       case 6: d = DiaSemana.SABADO;
               break;
       case 7: d = DiaSemana.DOMINGO;
               break;
}
```

Ejemplo 2: Igual que en el ejemplo anterior pero utilizando if anidados.

```
Scanner sc = new Scanner(System.in);
int valor;
DiaSemana d;
do {
    System.out.print("Introduce día de la semana del 1(Lunes) al 7(Domingo): ");
    valor = sc.nextInt();
} while (valor < 1 || valor > 7);
if(valor == 1){
   d = DiaSemana.LUNES;
}else if(valor == 2){
         d = DiaSemana.MARTES;
}else if(valor == 3){
         d = DiaSemana.MIERCOLES;
}else if(valor == 4){
         d = DiaSemana.JUEVES;
}else if(valor == 5){
         d = DiaSemana.VIERNES;
}else if(valor == 6){
         d = DiaSemana.SABADO;
}else{
      d = DiaSemana.DOMINGO;
}
```

Ejemplo 3: Leer un String que coincida **exactamente** con el valor de una constante del enum y utilizar el método **valueOf** para asignar ese valor a la variable enum.

```
Scanner sc = new Scanner(System.in);
DiaSemana d;
String dia;
System.out.print("Introduce día de la semana (LUNES, MARTES, ...): ");
dia = sc.nextLine();
d = DiaSemana.valueOf(dia.toUpperCase());
```

Si se hace de esta forma debemos asegurarnos que se ha introducido por teclado un valor que coincide con una constante. Si el String introducido no coincide con ninguna constante del enum, el método *valueOf* provocará un error de ejecución.

Ejemplo 4: Utilizando el método **values**. Este método devuelve un array con todas las constantes del enum. En el caso del enum DiaSemana el método *values* devolverá el siguiente array de tipo DiaSemana:

LUNES	MARTES	MIERCOLES	JUEVES	VIERNES	SABADO	DOMINGO
0	1	2	3	4	5	6

Se lee un valor entero y se utiliza como índice del array para obtener la constante.

```
Scanner sc = new Scanner(System.in);
DiaSemana d;
int valor;
System.out.print("Introduce día de la semana del 1(Lunes) al 7(Domingo) ");
valor = sc.nextInt();
d = DiaSemana.values()[valor - 1];
```

Esta es la mejor opción cuando el enum tiene muchas constantes. Nos evitamos hacer un switch o una cadena de if anidados demasiado grandes.

Uso avanzado de tipos enumerados

Se pueden declarar constantes de un enum relacionadas con un valor. Para ello la clase enum debe tener un atributo para cada valor relacionado con la constante y un constructor para asignar los valores. Esos valores se pasan al constructor cuando se crea la constante.

El constructor debe ser **private**. No puede ser público ya que no se pueden crear más objetos del enum.

Las constantes se deben definir primero, antes que cualquier otro atributo o método. Cuando un enum tiene atributos o métodos la lista de constantes debe acabar con punto y coma.

Los valores relacionados con las constantes se obtienen mediante el método get correspondiente.

Este tipo de enum no debe contener métodos set ya que los valores relacionados con las constantes se asignan en el constructor cuando se crean y ya no se pueden modificar.

Ejemplo:

```
//Declaración del enum
public enum Precios {
     PRECIONORMAL(100), PRECIOVIP(80);  //los valores se pasan al constructor
     private double precio;
     private Precios(double p){
         precio = p;
     }
     public double getPrecio() {
         return precio;
     }
}
```

```
//Clase principal
public class Enum3 {
    public static void main(String[] args) {
        Precios p = Precios.PRECIOVIP;     //precio = 80
        System.out.println(p.getPrecio()); //muestra 80
    }
}
```

Ejemplo:

Planetas es un tipo enum que representa los planetas del sistema solar.

Se define con los atributos masa y radio.

Cada constante del enum se declara con valores para los parámetros masa y radio.

Además del constructor, el enum tiene métodos que devuelven la gravedad y peso en cada planeta.

El programa lee el valor de un peso en la Tierra y calcula el peso equivalente en cada planeta.

```
//Enum Planetas
public enum Planetas {

    MERCURIO (3.303e+23, 2.4397e6),
    VENUS   (4.869e+24, 6.0518e6),
    TIERRA   (5.976e+24, 6.37814e6),
    MARTE    (6.421e+23, 3.3972e6),
    JUPITER (1.9e+27,   7.1492e7),
    SATURNO  (5.688e+26, 6.0268e7),
    URANO  (8.686e+25, 2.5559e7),
    NEPTUNO (1.024e+26, 2.4746e7);

    private final double masa;   // en kilogramos
    private final double radio;  // en metros

    public static final double G = 6.67300E-11; //constante gravitacional universal

    private Planetas(double masa, double radio) {
        this.masa = masa;
        this.radio = radio;
    }

    public double gravedad() {
        return G * masa / (radio * radio);
    }

    public double peso(double n) {
        return n * gravedad();
    }
}
//Clase Principal
public class Enum4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double pesoTerrestre, masaTerrestre;
        System.out.print("Introduce peso: ");
        pesoTerrestre = sc.nextDouble();
        masaTerrestre = pesoTerrestre/Planetas.TIERRA.gravedad();
        for (Planetas p : Planetas.values())
           System.out.printf("Su peso en %s es %.2f%n", p, p.peso(masaTerrestre));
    }
}
```