CONSTRUCTORES

Un constructor es un método que se ejecuta automáticamente cuando se crea un objeto de una clase. Sirve para inicializar los miembros de una clase.

El constructor tiene el mismo nombre que la clase. Cuando se define no se puede especificar un valor de retorno, nunca devuelve un valor. Sin embargo, puede tomar cualquier numero de argumentos.

Reglas

1. El constructor tiene el mismo nombre que la clase.
2. Puede tener cero o más argumentos.
3. No tiene tipo de retorno.

Ejemplo

La clase ***Rectángulo*** tiene un constructor con 4 parámetros.

```
Public class Rectangulo
{
  private int izdo;
  private int superior;
  private int dcha;
  private int inferior;
//constructor
public Rectangular (inti z, int sr, int d, int inf)
{
  izdo = iz;
  superior = sr;
  dcha = d;
  inferior = inf;
}
 // definiciones de otros métodos miembros
}
```

Al crear un objeto, se pasan los valores de los argumentos al constructor con la misma sintaxis que la de la llamada a un método. Por ejemplo:

```
Rectangulo Rect = new Rectangulo(25, 25,75, 75);
```

Se ha creado una instancia de ***Rectángulo*** pasando valores concretos al constructor de la clase, y de esa forma queda inicializado.

Constructor por defecto

un constructor que no tiene parámetros se le llama ***constructor por defecto***. Un constructor por defecto normalmente inicializa los miembros dato de la clase con valores por defecto.

> ***Reglas***
>
> Java crea automáticamente un constructor por defecto cuando no existen otros constructores. Tal constructor inicializa las variables de tipo numérico (**int**, **float**, …) a cero, las variables de tipo ***boolean*** a ***true*** y las referencias a ***null***.

Ejemplo

*el constructor por defecto inicializa **x** e **y** a **0**.*

```
Public class Punto
{
  private int x;
  private int y;
  public Punto() {    // constructor por defecto
   x = 0;
   y = 0;
  }
}
```

Cuando se crea un objeto ***Punto*** sus miembros dato se inicializan a 0.

```
Punto P1 = new Punto();     // P1.x == 0, P1.y == 0
```

Precaución

Tenga cuidado con la escritura de una clase con solo un constructor con argumentos. Si se omite un constructor sin argumento, no será posible utilizar el constructor por defecto.

La definición `NomClase c = new NomClase()` no será posible.

Constructor sobrecargado

Al igual que se puede sobrecargar un método de una clase, también se puede sobrecargar el constructor de una clase, De hecho, los constructores sobrecargados son bastante frecuentes y proporcionan diferentes alternativas para inicializar objetos.

Regla

Para prevenir a los usuarios de la clase de crear un objeto sin parámetros, se puede (1) omitir el constructor por defecto, o bien (2) hacer el constructor privado.

Ejemplo

La clase EquipoSonido se define con tres constructores: uno por defecto, otro con un argumento de tipo cadena y el tercero con tres argumentos.

```
Public class EquipoSonido

{

  private int potencia;

  private int voltios;

  private int numCd;

  private String marca;

public EquipoSonido ()   //constructor por defecto

{

  marca = "sin marca";

  System.out.println("Constructor por defecto");

}

public EquipoSonido (String mt)   //constructor con argumento

{

  marca = mt;

  System.out.println("Constructor con argumento cadena");

}

public EquipoSonido (String mt, int p, int v)   //constructor con argumento

{

  marca = mt;

  potencia = p;

  voltios = V;

  numCd = 20;

  System.out.println("Constructor con 3 argumento");

}

Public double factura(){…}

}
```

La instanciación de un objeto ***EquipoSonido*** puede hacerse llamando a cualquier constructor:

```
EquipoSonido rt, gt, ht;  // Define 3 referencias
rt = new EquipoSonido(); // llamada al constructor por defecto
gt = new EquipoSonido("JULAT");
ht = new EquipoSonido("PARTOLA",35,220);
```