Universidad Técnica Particular de loja
Ingeniería en Computación

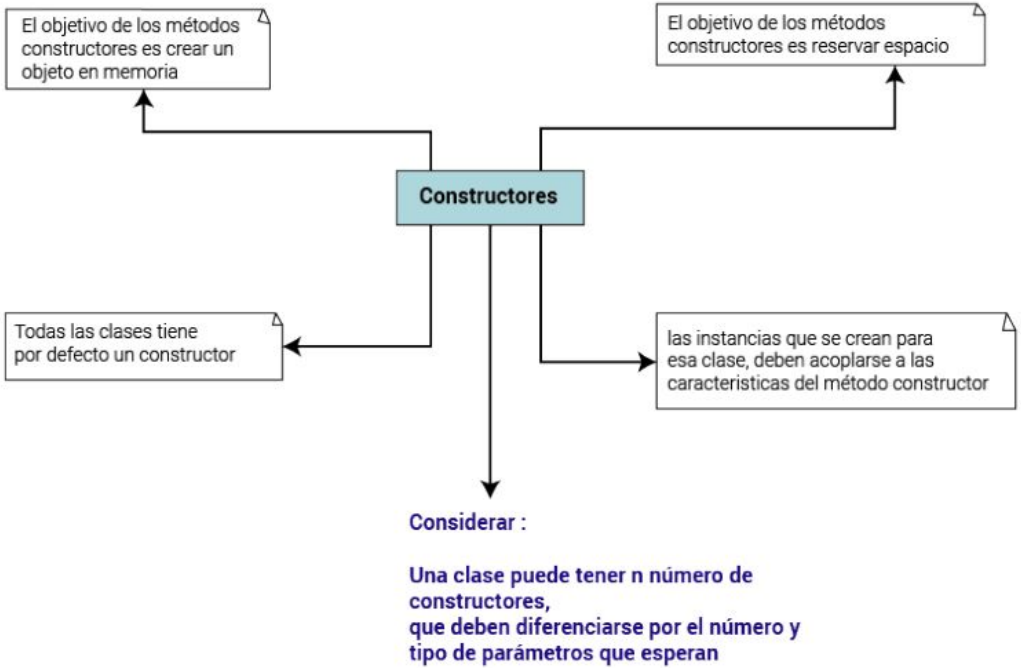
Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Manejo de constructores

René Rolando Elizalde Solano
rrelizalde@utpl.edu.ec
Abril - Agosto 2020

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores



Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores

- Método constructor.
- Java tiene un constructor por defecto.
- Se ejecuta con la palabra reservada new cuando se requiere indicar en miniespecificación o lenguajes como Java.
- Inicializa valores para los atributos de la clase.

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores
Java

- En la clase Ejecutor se crea un objeto llamado fac.
- fac: es un objeto de tipo FacturaTelefonica.
- fac: se crea como objeto a través de la sentenica new FacturaTelefonica(); donde:
  - **new**, es la palabra reservada para la creación de un objeto.
  - **FacturaTelefonica()**, hace referencia al constructor de la clase de la cual se necesita crear el objeto; atención con los paréntesis.

```
6   package paqueteuno;
7
8   public class Ejecutor {
9       public static void main(String[] args) {
10          // Se crea un objeto
11          // haciendo referencia al constructor por defecto
12          FacturaTelefonica fac = new FacturaTelefonica();
13      }
14  }
15
```

```
6   package paqueteuno;
7
8   public class FacturaTelefonica {
9
10      private String numeroTelefono;
11      private double minutosMes;
12      private double valorMinuto;
13      private double valorFactura;
14
15
16  }
```

Pero …
En la clase FacturaTelefonica
¿Dónde está el constructor ?

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores
Java

- Cuando la clase no tiene especificado un constructor y se crea un objeto para dicha clase desde otra clase, se llama (se hace uso) al constructor por defecto.

```
6   package paqueteuno;
7
8   public class FacturaTelefonica {
9
10      private String numeroTelefono;
11      private double minutosMes;
12      private double valorMinuto;
13      private double valorFactura;
14
15
16  }
```

```
6   package paqueteuno;
7
8   public class Ejecutor {
9       public static void main(String[] args) {
10          // Se crea un objeto
11          // haciendo referencia al constructor por defecto
12          FacturaTelefonica fac = new FacturaTelefonica();
13      }
14  }
15
```

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores

Java

- En la clase Ejecutor se crea un objeto a través del constructor por defecto de la clase FacturaTelefonica.
- Con el objeto fac, se presenta en pantalla los valores de cada atributo del objeto (con los métodos obtener.

```
6   package paquetedos;
7
8   public class Ejecutor {
9       public static void main(String[] args) {
10          // Se crea un objeto
11          // haciendo referencia al constructor por defecto
12          FacturaTelefonica fac = new FacturaTelefonica();
13          System.out.printf("Número telefónico: %s\n"
14                  + "Minutos mes: %.2f\nValor minuto: %.2f\n"
15                  + "Valor factura: %.2f\n", fac.obtenerNumeroTelefono(),
16                  fac.obtenerMinutosMes(),
17                  fac.obtenerValorMinuto(),
18                  fac.obtenerValorFactura());
19      }
20  }
```

```
6   package paquetedos;
7
8   public class FacturaTelefonica {
9
10      private String numeroTelefono;
11      private double minutosMes;
12      private double valorMinuto;
13      private double valorFactura;
14
15      public void establecerNumeroTelefono(String c) {
16          numeroTelefono = c;
17      }
18
19      public void establecerMinutosMes(double c) {
20          minutosMes = c;
21      }
22
23      public void establecerValorMinuto(double c) {
24          valorMinuto = c;
25      }
26
27      // public void establecerValorFactura(){
28      public void calcularValorFactura() {
```

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores Java

- ¿Cuál es la salida por pantalla?. Atención hemos usado el constructor por defecto.

```java
package paquetedos;

public class Ejecutor {
    public static void main(String[] args) {
        // Se crea un objeto
        // haciendo referencia al constructor por defecto
        FacturaTelefonica fac = new FacturaTelefonica();
        System.out.printf("Número telefónico: %s\n"
                + "Minutos mes: %.2f\nValor minuto: %.2f\n"
                + "Valor factura: %.2f\n", fac.obtenerNumeroTelefono(),
                fac.obtenerMinutosMes(),
                fac.obtenerValorMinuto(),
                fac.obtenerValorFactura());
    }
}
```

```java
package paquetedos;

public class FacturaTelefonica {

    private String numeroTelefono;
    private double minutosMes;
    private double valorMinuto;
    private double valorFactura;

    public void establecerNumeroTelefono(String c) {
        numeroTelefono = c;
    }

    public void establecerMinutosMes(double c) {
        minutosMes = c;
    }

    public void establecerValorMinuto(double c) {
        valorMinuto = c;
    }

    // public void establecerValorFactura(){
    public void calcularValorFactura() {
```

```
run:
Número telefónico: null
Minutos mes: 0,00
Valor minuto: 0,00
Valor factura: 0,00
BUILD SUCCESSFUL (total
```

- A pesar de no asignar en ningún momento valores a los atributos del objeto; los método obtener, devuelven los valores por defecto de cada atributo.
- String: null
- int: 0
- double: 0.0
- boolean: False

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores Java

- Hasta ahora hemos asignado valores a los atributos de los objetos mediante los métodos establecer (recuerden, los atributos tienen visibilidad privada).
- Pero … existe otra forma? Sí, en función del concepto de un constructor; se puede crear un objetos a través de personalizar (sobrecargar) el constructor de una clase.

```
8   public class FacturaTelefonica {
9
10      private String numeroTelefono;
11      private double minutosMes;
12      private double valorMinuto;
13      private double valorFactura;
14
15      public FacturaTelefonica(){
16          numeroTelefono = "123456789";
17          minutosMes = 100;
18          valorMinuto = 0.25;
19
20      }
```

- Inicializa valores para los atributos de la clase.

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores
Java

Palabra reservada para crear el constructor y su visibilidad

Nombre del constructor; debe tener el mismo nombre que la clase.

Conjunto de argumentos del constructor

```
14
15      public FacturaTelefonica(){
16          numeroTelefono = "123456789";
17          minutosMes = 100;
18          valorMinuto = 0.25;
19
20      }
```

Asignaciones o lógica para el constructor

Fin del constructor

Inicio del constructor

- Para la declaración del método constructor de una clase se usa el mismo nombre de la clase **seguido de un paréntesis ()**.
  - public Factura()
- En el método constructor del ejemplo se realizan asignaciones de valores específicos a los datos (atributos) para un objeto.
- Atención: el número de parámetros del constructor puede ser cero, uno, dos, etc; depende la lógica que se aplique al constructor.

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores

Java

- ¿Cuál es la salida por pantalla?. Atención hemos usado el constructor personalizado sin argumentos; ya dejamos de usar el constructor por defecto.

```
6   package paquetetres;
7
8   public class Ejecutor {
9       public static void main(String[] args) {
10          // Se crea un objeto
11          // haciendo referencia al constructor por defecto
12          FacturaTelefonica fac = new FacturaTelefonica();
13          System.out.printf("Número telefónico: %s\n"
14                  + "Minutos mes: %.2f\nValor minuto: %.2f\n"
15                  + "Valor factura: %.2f\n", fac.obtenerNumeroTelefono(),
16                  fac.obtenerMinutosMes(),
17                  fac.obtenerValorMinuto(),
18                  fac.obtenerValorFactura());
19      }
20  }
```

```
8   public class FacturaTelefonica {
9
10      private String numeroTelefono;
11      private double minutosMes;
12      private double valorMinuto;
13      private double valorFactura;
14
15      public FacturaTelefonica(){
16          numeroTelefono = "123456789";
17          minutosMes = 100;
18          valorMinuto = 0.25;
19
20      }
```

```
run:
Número telefónico: 123456789
Minutos mes: 100,00
Valor minuto: 0,25
Valor factura: 0,00
BUILD SUCCESSFUL (total time:
```

- El valor del atributo Valor factura es 0.0, debido a que no le hemos asignado valores; en el constructor no se asigna valores a ese atributo.

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores Java

```
8   public class FacturaTelefonica {
9
10      private String numeroTelefono;
11      private double minutosMes;
12      private double valorMinuto;
13      private double valorFactura;
14
15      public FacturaTelefonica(String numTel, double mMes, double vMinuto){
16          numeroTelefono = numTel;
17          minutosMes = mMes;
18          valorMinuto = vMinuto;
19
20      }
21
```

- Se ha utilizado constructores por defecto.
- De igual manera constructores sin argumentos.
- Se puede usar constructores con argumentos? Sí.
  - Esos argumentos sirven para asignar valores a los atributos.

- El método constructor de la clase tiene 3 argumentos: uno de tipo cadena, dos de tipo double.
- En la lógica del constructor se asigna valores a los atributos:
  - numeroTelefono toma el valor del argumento numTel
  - minutosMes toma el valor del argumento mMes
  - valorMinuto toma el valor del argumento vMinuto

Atención:

- **Para los posibles objetos ya no se puede usar el constructor por defecto de la clase.**
- **Todos los objetos de tipo FacturaTelefonica deben hacer referencia al constructor y sus argumentos.**

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores Java

```
Output - EjemploConstructores (run)
Exception in thread "main" java.lang.RuntimeException: Uncom
  required: java.lang.String,double,double
  found: no arguments
  reason: actual and formal argument lists differ in length
        at paquetecuatro.Ejecutor.main(Ejecutor.java:12)
```

```
8   public class FacturaTelefonica {
9
10      private String numeroTelefono;
11      private double minutosMes;
12      private double valorMinuto;
13      private double valorFactura;
14
15      public FacturaTelefonica(String numTel, double mMes, double vMinuto){
16          numeroTelefono = numTel;
17          minutosMes = mMes;
18          valorMinuto = vMinuto;
19
20      }
21
```

```
6   package paquetecuatro;
7
8   public class Ejecutor {
9       public static void main(String[] args) {
10          // Se crea un objeto
11          // haciendo referencia al constructor por defecto
12          FacturaTelefonica fac = new FacturaTelefonica();
13          System.out.printf("Número telefónico: %s\n"
14                  + "Minutos mes: %.2f\nValor minuto: %.2f\n"
15                  + "Valor factura: %.2f\n", fac.obtenerNumeroTelefono(),
16                  fac.obtenerMinutosMes(),
17                  fac.obtenerValorMinuto(),
18                  fac.obtenerValorFactura());
19      }
```

Estamos llamando a un constructor sin argumentos, pero la clase tiene un constructor con argumentos.

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Constructores

Java

```java
public class FacturaTelefonica {

    private String numeroTelefono;
    private double minutosMes;
    private double valorMinuto;
    private double valorFactura;

    public FacturaTelefonica(String numTel, double mMes, double vMinuto){
        numeroTelefono = numTel;
        minutosMes = mMes;
        valorMinuto = vMinuto;

    }
```

```java
package paquetecinco;

public class Ejecutor {
    public static void main(String[] args) {
        // Se crea un objeto que tiene un contructor
        // con argumentos
        String numero = "123456789";
        double minutos = 100;
        double valorMinutos = 0.25;

        FacturaTelefonica fac = new FacturaTelefonica(numero, minutos,
                valorMinutos);
        System.out.printf("Número telefónico: %s\n"
                + "Minutos mes: %.2f\nValor minuto: %.2f\n"
                + "Valor factura: %.2f\n", fac.obtenerNumeroTelefono(),
                fac.obtenerMinutosMes(),
                fac.obtenerValorMinuto(),
                fac.obtenerValorFactura());
    }
}
```

```
Output - EjemploConstructores (run)
run:
Número telefónico: 123456789
Minutos mes: 100,00
Valor minuto: 0,25
Valor factura: 0,00
BUILD SUCCESSFUL (total time: 1 second)
```

- Estamos llamando a un constructor y se establece la correspondencia que requiere el constructor.
- Para establecer los valores al objeto no se usa los métodos establecer; le delegamos la función al constructor.

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

```
8   public class FacturaTelefonica {
9
10      private String numeroTelefono;
11      private double minutosMes;
12      private double valorMinuto;
13      private double valorFactura;
14
15      public FacturaTelefonica(){
16          numeroTelefono = "234567891";
17          minutosMes = 200;
18          valorMinuto = 0.15;
19
20      }
21      public FacturaTelefonica(String numTel, double mMes, double vMinuto){
22          numeroTelefono = numTel;
23          minutosMes = mMes;
24          valorMinuto = vMinuto;
25
26      }
```

¿Se puede tener dos o más constructores? **Sí**.

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Universidad Técnica Particular de loja
Ingeniería en Computación

Solución de la clase Ejecutor

```
9   public static void main(String[] args) {
10      // Se crea un objeto haciendo referencia al constructor sin
11      // argumentos
12      FacturaTelefonica fac = new FacturaTelefonica();
13      System.out.println("Objeto sin argumentos\n");
14      System.out.printf("Número telefónico: %s\n"
15              + "Minutos mes: %.2f\nValor minuto: %.2f\n"
16              + "Valor factura: %.2f\n", fac.obtenerNumeroTelefono(),
17              fac.obtenerMinutosMes(),
18              fac.obtenerValorMinuto(),
19              fac.obtenerValorFactura());
20
21      // Se crea un objeto haciendo referencia al constructor
22      // con argumentos de la clase
23      String numero = "123456789";
24      double minutos = 100;
25      double valorMinutos = 0.25;
26
27      FacturaTelefonica fac2 = new FacturaTelefonica(numero, minutos,
28              valorMinutos);
29      System.out.println("\nObjeto con argumentos\n");
30      System.out.printf("Número telefónico: %s\n"
31              + "Minutos mes: %.2f\nValor minuto: %.2f\n"
32              + "Valor factura: %.2f\n", fac2.obtenerNumeroTelefono(),
33              fac2.obtenerMinutosMes(),
34              fac2.obtenerValorMinuto(),
35              fac2.obtenerValorFactura());
36      }
37  }
```

```
15  public FacturaTelefonica(){
16      numeroTelefono = "234567891";
17      minutosMes = 200;
18      valorMinuto = 0.15;
19
20  }
```

```
22  public FacturaTelefonica(String numTel, double mMes,
23          double vMinuto) {
24
25      numeroTelefono = numTel;
26      minutosMes = mMes;
27      valorMinuto = vMinuto;
28
29  }
```

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Universidad Técnica Particular de loja
Ingeniería en Computación

Solución de la clase Ejecutor

```
9   public static void main(String[] args) {
10      // Se crea un objeto haciendo referencia al constructor sin
11      // argumentos
12      FacturaTelefonica fac = new FacturaTelefonica();
13      System.out.println("Objeto sin argumentos\n");
14      System.out.printf("Número telefónico: %s\n"
15              + "Minutos mes: %.2f\nValor minuto: %.2f\n"
16              + "Valor factura: %.2f\n", fac.obtenerNumeroTelefono(),
17              fac.obtenerMinutosMes(),
18              fac.obtenerValorMinuto(),
19              fac.obtenerValorFactura());
20
21      // Se crea un objeto haciendo referencia al constructor
22      // con argumentos de la clase
23      String numero = "123456789";
24      double minutos = 100;
25      double valorMinutos = 0.25;
26
27      FacturaTelefonica fac2 = new FacturaTelefonica(numero, minutos,
28              valorMinutos);
29      System.out.println("\nObjeto con argumentos\n");
30      System.out.printf("Número telefónico: %s\n"
31              + "Minutos mes: %.2f\nValor minuto: %.2f\n"
32              + "Valor factura: %.2f\n", fac2.obtenerNumeroTelefono(),
33              fac2.obtenerMinutosMes(),
34              fac2.obtenerValorMinuto(),
35              fac2.obtenerValorFactura());
36  }
37 }
```

```
run:
Objeto sin argumentos

Número telefónico: 234567891
Minutos mes: 200,00
Valor minuto: 0,15
Valor factura: 0,00

Objeto con argumentos

Número telefónico: 123456789
Minutos mes: 100,00
Valor minuto: 0,25
Valor factura: 0,00
```

Unidad 2: Estructura y creación de programas en Programación Orientada a Objetos

Universidad Técnica Particular de loja
Ingeniería en Computación

Solución de la clase Ejecutor; valor factura ya no debe ser cero.

```
12          FacturaTelefonica fac = new FacturaTelefonica();
13          // se llama al método que calcula el valor de la factura
14          fac.calcularValorFactura();
15          System.out.println("Objeto sin argumentos\n");
16          System.out.printf("Número telefónico: %s\n"
17                  + "Minutos mes: %.2f\nValor minuto: %.2f\n"
18                  + "Valor factura: %.2f\n", fac.obtenerNumeroTelefono(),
19                  fac.obtenerMinutosMes(),
20                  fac.obtenerValorMinuto(),
21                  fac.obtenerValorFactura());
```

```
25          String numero = "123456789";
26          double minutos = 100;
27          double valorMinutos = 0.25;
28
29          FacturaTelefonica fac2 = new FacturaTelefonica(numero, minutos,
30                  valorMinutos);
31          // se llama al método que calcula el valor de la factura
32          fac2.calcularValorFactura();
33          System.out.println("\nObjeto con argumentos\n");
34          System.out.printf("Número telefónico: %s\n"
35                  + "Minutos mes: %.2f\nValor minuto: %.2f\n"
36                  + "Valor factura: %.2f\n", fac2.obtenerNumeroTelefono(),
37                  fac2.obtenerMinutosMes(),
38                  fac2.obtenerValorMinuto(),
39                  fac2.obtenerValorFactura());
40      }
```

```
Objeto sin argumentos

Número telefónico: 234567891
Minutos mes: 200,00
Valor minuto: 0,15
Valor factura: 30,00

Objeto con argumentos

Número telefónico: 123456789
Minutos mes: 100,00
Valor minuto: 0,25
Valor factura: 25,00
```

Gracias