MÓDULO POO

Constructores y sobrecarga

Autores:
Alejandro Héctor Gonzalez
Silvana Lis Gallo
Junio 2021

RESUMEN

En esta clase trabajaremos el concepto de constructores y sobrecarga. Se revisa como un objeto puede utilizar a otro objeto y cómo diferenciarlos con la cláusula “this”

Palabras clave

Constructor, this, relaciones entre objetos, sobrecarga

Instanciar e iniciar objeto

- Hasta ahora, nuestro main ...

```
public class DemoLibro {

public static void main(String[] args) {
        Libro libro = new Libro();
        libro.setTitulo("Java: A Beginner's Guide");
        libro.setEditorial("Mcgraw-Hill");
        libro.setAñoEdicion(2014);
        libro.setPrimerAutor("Herbert Schildt");
        libro.setISBN("978-0071809252");
        libro.setPrecio(21.72);
        …

    }
}
```

Generar una clase para representar libros. Un Libro se caracteriza por: título, nombre del primer autor, editorial, año de edición, ISBN, precio
El libro debe saber:

- Devolver el valor de cada atributo.
- Modificar el valor de cada atributo.
- Devolver su representación en formato String.
  Repr. *"Java: A Beginner's Guide por Herbert Schildt - 2014 - ISBN: 978-0071809252"*

Declaración de constructores.

- Se **ejecuta** tras alocar el objeto e inicializar las v.i. (por defecto o explícitamente).
- Objetivo: **inicialización de v.i.**
- Sintaxis

```
public NombreClase( lista de parámetros formales ){
    /* Código */
}
```

- **Si la clase no declara** ningún constructor, Java incluye uno sin parámetros y sin código (*constructor nulo*).
- Instanciación de objeto:

```
NombreClase objeto= new NombreClase(lista de parámetros actuales);
```

Ejemplo (Hasta ahora) `Libro miLibro = new Libro();` //Invoca al *constructor nulo*

Declaración de constructores. Ejemplo.

```
public class Libro {
   private String titulo;
   private String primerAutor;
   private String editorial;
   private int añoEdicion;
   private String ISBN;
   private double precio;
   …
```

```
   public Libro(  String unTitulo,  String unaEditorial,
                  int unAñoEdicion,  String unPrimerAutor,
                  String unISBN, double unPrecio){
      titulo = unTitulo;
      editorial = unaEditorial;
      añoEdicion= unAñoEdicion;
      primerAutor = unPrimerAutor;
      ISBN =  unISBN;
      precio = unPrecio;
   }

   ….

}
```

Declaración de constructores. Ejemplo.

- Ejemplo instanciación (en main)

```
Libro libro1= new  Libro( "Java: A Beginner's Guide",  "Mcgraw-Hill",
                          2014,  "Herbert Schildt",
                          "978-0071809252", 21.72);
```

- ¿Funciona ahora? Libro libro = new Libro();

Si el programador generó un constructor,
Java NO incluye el constructor nulo.

Declaración de constructores. Sobrecarga. Ejemplo.

- Puede haber varios constructores para la clase (sobrecarga).
- Java identifica cuál está siendo invocado por el número y tipo de sus parámetros.
- *Por defecto quiero que el libro tenga año de edición 2015 y precio 100 => Otro constructor*

```
public class Libro {
   private String titulo;
   private String primerAutor;
   private String editorial;
   private int añoEdicion;
   private String ISBN;
   private double precio;

    public Libro(  String unTitulo,  String unaEditorial,
    int unAñoEdicion,  String unPrimerAutor, String unISBN, double unPrecio){
        titulo = unTitulo;
        editorial = unaEditorial;
        añoEdicion= unAñoEdicion;
        primerAutor = unPrimerAutor;
        ISBN =  unISBN;
        precio = unPrecio;
    }

    public Libro(  String unTitulo,  String unaEditorial, String
unPrimerAutor, String unISBN){
        titulo = unTitulo;
        editorial = unaEditorial;
        añoEdicion= 2015;
        primerAutor = unPrimerAutor;
        ISBN =  unISBN;
        precio = 100;
    }

    public Libro(){

    }
    …
}
```

3 constructores distintos

Declaración de constructores. Sobrecarga. Ejemplo.

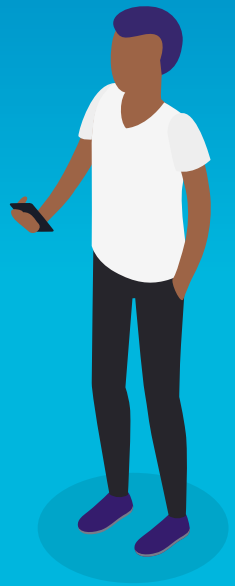
```
public class DemoConstructoresLibro {
public static void main(String[] args) {
        Libro libro1= new  Libro( "Java: A Beginner's Guide",  "Mcgraw-Hill", 2014,
                        "Herbert Schildt", "978-0071809252", 21.72);
        Libro libro2= new Libro("Learning Java by Building Android Games",
                        "CreateSpace Independent Publishing",
                        "John Horton", "978-1512108347");
        System.out.println(libro1.toString());
        System.out.println(libro2.toString());
        System.out.println("Precio del libro2: " +libro2.getPrecio());
        System.out.println("Año edición del libro2: " +libro2.getAñoEdicion());
        Libro libro3= new Libro();
    }

}
```

¿Funciona?

Probar ejemplo en carpeta Cosntructores

- Interacción entre objetos



Interacción entre objetos. Ejemplo

- Los objetos **cooperan** (enviándose mensajes) para llevar a cabo una tarea común ...
- Ej: Hasta ahora nuestros libros consideran al primer autor como un String.

¿Y si el autor fuese un objeto instancia de la clase Autor?

¿Qué modificaciones debo hacer en el código?

Diagrama de clases

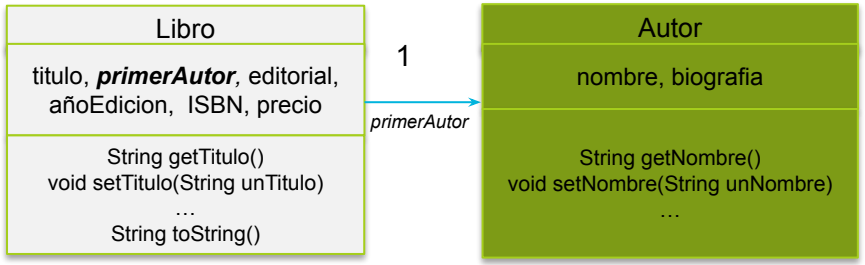
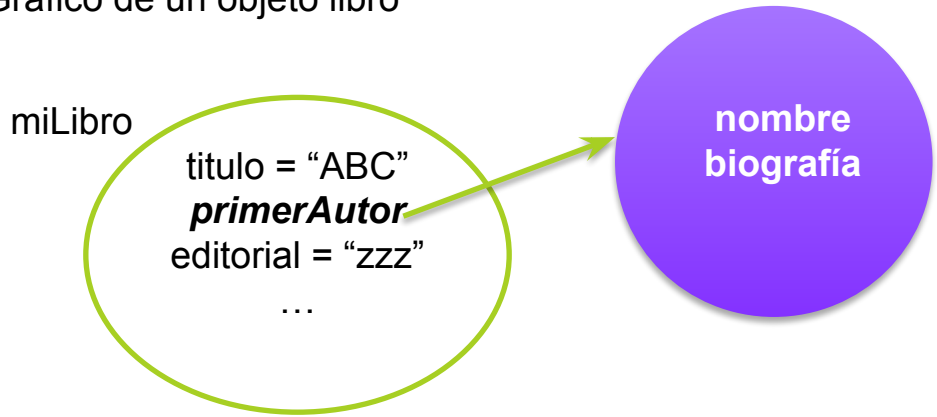


Gráfico de un objeto libro



Interacción entre objetos

- Ejemplo: ¿qué pasos seguiría en el prog. ppal. para imprimir el nombre del autor del libro?

Diagrama de clases

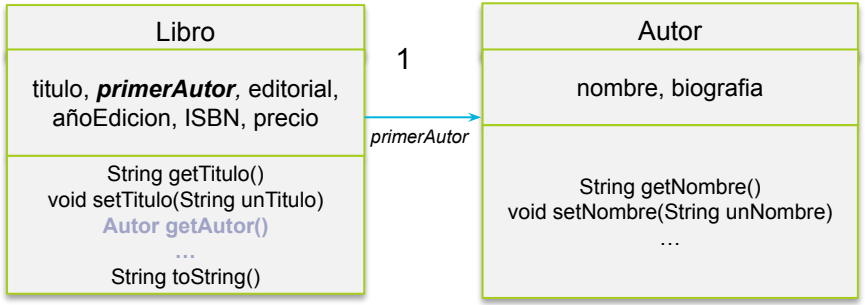
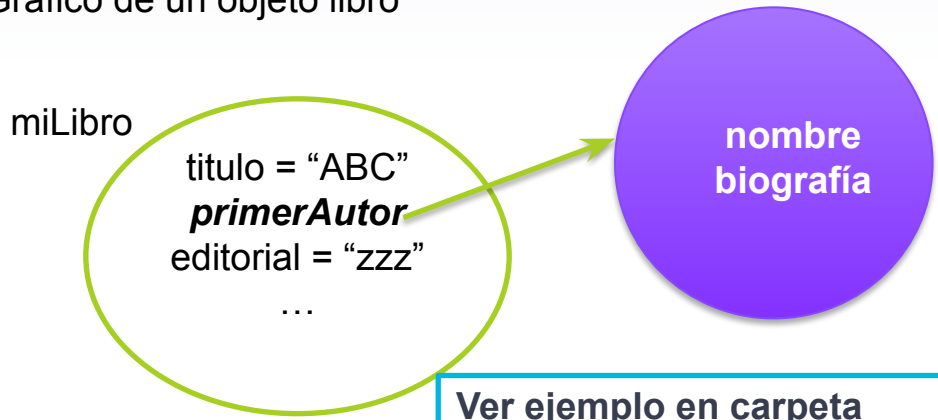


Gráfico de un objeto libro



Ver ejemplo en carpeta RelacionesEntreObjetos

1 – Pedirle al objeto libro que me devuelva el autor
2 – Una vez que obtengo el autor le pido a ese objeto que me devuelva su nombre

Referencia “this”

La referencia this

- Dentro de un *método de instancia* o de un *constructor*, la referencia *this* representa al objeto que recibió el mensaje o el objeto que está siendo instanciado respectivamente.
- Uso:

a) Los parámetros del método/constructor que se ejecuta actualmente tienen el mismo nombre que las variables de instancia del objeto. Para referirse a las variables de la instancia se utiliza *this.nombreVariableInstancia*

```
public class Libro {
    private String titulo;
    private int paginas;
    private String editorial;
    private int añoEdicion;
    private String idioma;
    private Autor primerAutor;
    private String ISBN;
    private double precio;
    private int cantidadEnStock;
```

```
public Libro(  String titulo, int paginas,  String editorial,
  int añoEdicion, String idioma,  Autor primerAutor,
  String ISBN, double precio,  int cantidadEnStock){
     this.titulo= titulo;
     this.paginas= paginas;
     this.editorial= editorial;
     this.añoEdicion= añoEdicion;
     this.idioma= idioma;
     this.primerAutor= primerAutor;
     this.ISBN= ISBN;
     this.precio= precio;
     this.cantidadEnStock= cantidadEnStock;
  }
```

```
public void setTitulo(String titulo){
   this.titulo = titulo;
}
```

Ver en carpeta=> UsandoThis
=> Autor.java
=> Libro.java
=> DemoUsandoThis.java

La referencia this

- Uso:

b) El objeto receptor del mensaje o el objeto que está siendo construido debe enviarse mensajes a sí mismo, ej. para desencadenar la ejecución de métodos más simples. Para enviarse un mensaje a sí mismo hacer this.nombreMetodo(parámetros)

```
public class Libro {
   …
   public Libro(  String titulo, int paginas,  String editorial, int añoEdicion, String idioma,
                  Autor primerAutor, String ISBN, double precio,  int cantidadEnStock){
      this.setTitulo(titulo);
      this.setPaginas(paginas);
      …
   }
  public String toString(){
     return (this.getTitulo() + " por " + this.getPrimerAutor().getNombre() + " - " +
             this.getAñoEdicion() + " -  ISBN: " + this.getISBN() );
   }
}
```

La referencia this

- Uso:

c) Invocar desde un constructor a otro, ej. para evitar repetir código. Para invocar a un segundo constructor hacer this(parámetros)

```
public Libro(  String titulo, int paginas,  String editorial,
  int añoEdicion, String idioma,  Autor primerAutor,
  String ISBN, double precio,  int cantidadEnStock){
     this.titulo= titulo;
     this.paginas= paginas;
     this.editorial= editorial;
     this.añoEdicion= añoEdicion;
     this.idioma= idioma;
     this.primerAutor= primerAutor;
     this.ISBN= ISBN;
     this.precio= precio;
     this.cantidadEnStock= cantidadEnStock;
  }
```

```
 public Libro(  String titulo, int paginas,  String editorial, Autor
primerAutor, String ISBN, double precio,  int cantidadEnStock){
        this.titulo = titulo;
        this.paginas = paginas;
        this.editorial = editorial;
        this.añoEdicion= 2015;
        this.idioma= "Inglés";
        this.primerAutor = primerAutor;
        this.ISBN =  ISBN;
        this.precio = precio;
        this.cantidadEnStock = cantidadEnStock;
    }
```

Código repetido: sería mejor invocar al 1er constructor

La referencia this

- Uso:

c) Invocar desde un constructor a otro, ej. para evitar repetir código. Para invocar a un segundo constructor hacer this(parámetros)

```
public Libro(  String titulo, int paginas,  String editorial,
   int añoEdicion, String idioma,  Autor primerAutor,
   String ISBN, double precio,  int cantidadEnStock){
      this.titulo= titulo;
      this.paginas= paginas;
      this.editorial= editorial;
      this.añoEdicion= añoEdicion;
      this.idioma= idioma;
      this.primerAutor= primerAutor;
      this.ISBN= ISBN;
      this.precio= precio;
      this.cantidadEnStock= cantidadEnStock;
   }
```

```
public Libro(  String titulo, int paginas,  String editorial, Autor
primerAutor, String ISBN, double precio,  int cantidadEnStock){

      this( titulo, paginas,  editorial, 2015, "inglés",
            primerAutor, ISBN, precio, cantidadEnStock);

   }
```

Restricción!!!: la invocación a otro constructor debe ser la primera línea de código