

Ejemplo: Detalle de un Libro con Java



📚 Guía: Manejo de Memoria en Java - Stack y Heap

◆ Introducción

En esta lección, exploraremos el manejo de memoria en Java, específicamente las diferencias entre **Stack** y **Heap**. 🧠📌

Para ello, crearemos un programa en Java que nos permitirá visualizar cómo se almacenan y manipulan diferentes tipos de variables.

🛠 Paso 1: Crear el archivo `DetalleLibro.java`

- ◆ Abrimos **IntelliJ IDEA** y creamos un nuevo archivo dentro de nuestro proyecto existente de Variables.
- 📁 **Ruta del archivo:** `src/main/java/DetalleLibro.java`

💻 Código inicial:

<https://www.globalmentoring.com.mx>

```
public class DetalleLibro {  
    public static void main(String[] args) {  
        // Detalle de un Libro  
    }  
}
```

❖ Paso 2: Declarar variables y su almacenamiento en memoria

- ◆ Vamos a definir cuatro variables para representar la información de un libro:

- 📖 `tituloLibro` (String) → Título del libro
- 📅 `anioPublicacion` (int) → Año de publicación
- ✅ `libroDisponible` (boolean) → Si está disponible
- 💰 `precio` (double) → Precio del libro

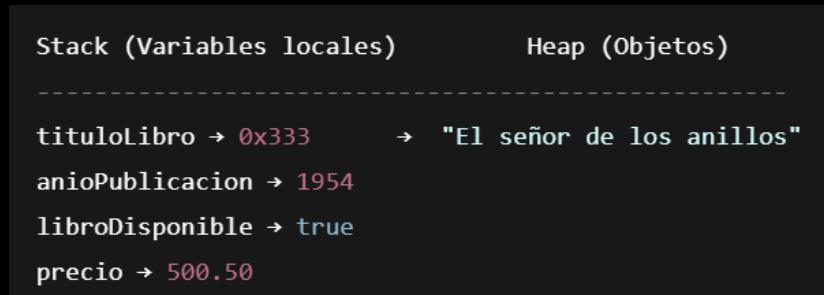
💻 Código agregado:

```
String tituloLibro = "El señor de los anillos";  
int anioPublicacion = 1954;  
boolean libroDisponible = true;  
double precio = 500.50;
```

- ◆ ¿Qué pasa en memoria?

- La **memoria Stack** almacena las variables locales (`tituloLibro`, `anioPublicacion`, `libroDisponible`, `precio`).
- La **memoria Heap** almacena el **objeto String**, ya que las cadenas en Java son objetos.

❖ Ejemplo gráfico en memoria:



❖ Paso 3: Imprimir los valores de las variables

- ◆ Ahora, imprimimos cada una de estas variables para visualizar sus valores en la consola.

💻 Código agregado:

```
System.out.println(tituloLibro);
System.out.println(anioPublicacion);
System.out.println(libroDisponible);
System.out.println(precio);
```

📌 Salida esperada en consola:

```
El señor de los anillos
1954
true
500.5
```

🚀 Aquí estamos accediendo a la memoria para recuperar y mostrar los valores almacenados.

❖ Paso 4: Modificar el valor de una variable (Cadenas en memoria Heap)

- ◆ Ahora cambiaremos el título del libro a una versión con mayúsculas en "Señor" y "Anillos":

💻 Código agregado:

```
tituloLibro = "El Señor de los Anillos";
System.out.println(tituloLibro);
```

- ◆ ¿Qué pasa en memoria?

- En la memoria **Heap**, Java NO sobrescribe el objeto original, sino que crea un **nuevo objeto** con la nueva cadena "El Señor de los Anillos".
- En la memoria **Stack**, la variable `tituloLibro` ahora apunta a la nueva dirección de memoria.

📌 Ejemplo gráfico en memoria:

Stack (Variables locales)	Heap (Objetos)
tituloLibro → 0x444	→ "El Señor de los Anillos"
	X 0x333 → "El señor de los anillos" (sin referencia, será eliminado)
anioPublicacion → 1954	
libroDisponible → true	
precio → 500.50	

📌 Salida esperada en consola:

El Señor de los Anillos

🚀 Aquí vemos cómo el recolector de basura de Java eliminará automáticamente el objeto anterior, ya que ninguna variable lo apunta.

Nosotros no podemos ejecutar directamente el recolector de basura, pero podemos darle la indicación a Java que cuando sea posible elimine los objetos que ya no están siendo referenciados por ninguna variable con la siguiente instrucción (Java decidirá cuándo ejecutarlo (no es inmediato ni garantizado)):

```
System.gc(); // Llama al recolector de basura para que se ejecute en cuanto se pueda
```

🎯 Código Final

```
public class DetalleLibro {
    public static void main(String[] args) {
        // Detalle de un Libro
        String tituloLibro = "El señor de los anillos";
        int anioPublicacion = 1954;
        boolean libroDisponible = true;
        double precio = 500.50;
        // Imprimir el valor de las variables
        System.out.println(tituloLibro);
        System.out.println(anioPublicacion);
        System.out.println(libroDisponible);
        System.out.println(precio);
        // Modificar el título de libro
        tituloLibro = "El Señor de los Anillos";
        System.out.println(tituloLibro);
    }
}
```

👉 Conclusión

- ◆ En esta lección hemos aprendido:
- ✓ Cómo se almacenan las variables en **Stack** y **Heap**.
- ✓ La diferencia entre **tipos primitivos** y **objetos String** en memoria.
- ✓ Cómo **modificar el valor de una variable** y qué sucede internamente en Java.

💡 Ejercicio recomendado:

✖ Intenta agregar más atributos, como el **nombre del autor** o el **número de páginas**, e imprime su contenido.

- ◆ ¡Sigue practicando y nos vemos en la siguiente lección! 🚀

¡Saludos! 🌟

Ing. Marcela Gamiño e Ing. Ubaldo Acosta

Fundadores de [GlobalMentoring.com.mx](https://www.globalmentoring.com.mx)