Construcción de Objetos (Constructores)

Programación y Diseño Orientado a Objetos (PDOO)

📋 Tabla de Contenidos

Constructores: Conceptos Generales

Los constructores son métodos esenciales en la **POO** que garantizan la **inicialización correcta** del estado interno de un objeto justo después de su reserva de memoria.

Características Fundamentales

Característica	Descripción
Propósito	Inicialización de **TODOS** los atributos de instancia
Diferencia Clave	No se encargan de la reserva de memoria (eso lo hace new), solo de la inicialización
Naturaleza	No son métodos de instancia normales y no especifican tipo de retorno

Clasificación por Tipo de Clase

Tipo de Clase	Lenguajes	Características
Clases-Plantilla	Java, C++	Mismo nombre que la clase. Se invocan con new
Clases-Objeto	Ruby	Nombre arbitrario o métodos de clase. Constructor llamado internamente por new

---

Constructores en Java

Características

Característica	Descripción
Nombre	Debe tener el **mismo nombre** que la clase
Sobrecarga	✅ Permite overloading (varios constructores con diferentes parámetros)
Constructor por Defecto	Si no se define ninguno, Java proporciona uno sin parámetros
Reutilización	Se puede llamar a otro constructor con this()

Ejemplo: Constructor y Reutilización

```
public class RestrictedPoint3D {
    // Atributos de instancia
    private int x, y, z;
    // Constructor Principal (con 3 parámetros)
    RestrictedPoint3D(int x, int y, int z) {
        // Se usa 'this.' para distinguir el atributo del parámetro
        this.x = restricToRange(x);
        this.y = restricToRange(y);
        this.z = restricToRange(z);
    }
    // Constructor Secundario (con 2 parámetros)
    RestrictedPoint3D(int x, int y) {
        // Reutiliza el Constructor Principal, asumiendo z = 0 por defecto
        this(x, y, 0);
    }
    // Método auxiliar (Restringe el valor al rango [0, 100])
    private int restricToRange(int a) {
        // Lógica de restricción (omitted for brevity)
        return a;
    }
}
```

Constructores en Ruby

Características

Característica	Descripción
Método Equivalente	El constructor es un método de instancia llamado initialize
Llamada	Es privado y llamado automáticamente por el método de clase new
Sobrecarga	❌ Ruby no permite sobrecargar métodos (incluyendo initialize)

Alternativas para Múltiples Constructores

1. Crear métodos de clase con nombres distintos (ej. `new_2D`, `new_3D`) que llamen a `initialize`
2. Usar parámetros con valores por defecto o parámetros nombrados en `initialize`

Ejemplo: Múltiples Constructores usando Métodos de Clase

```
class RestrictedPoint3D
  # ... atributos y método restric_to_range ...
  def initialize(x, y, z)
    # Inicialización de atributos de instancia
    @x = restric_to_range(x)
    @y = restric_to_range(y)
    @z = restric_to_range(z)
  end
  # Constructor alternativo 1 (Método de Clase)
  def self.new_3D(x, y, z)
    new(x, y, z) # Llama al new por defecto que llama a initialize
  end
  # Constructor alternativo 2 (Método de Clase)
  def self.new_2D(x, y)
    new(x, y, 0) # Llama al new por defecto, que llama a initialize con z=0
  end
  # Hacer que el 'new' por defecto sea privado para forzar el uso de new_3D/new_2D
  private_class_method :new
end
```

Ejemplo: Parámetros Nombrados con Valores por Defecto

```
class RestrictedPoint3D
  # Utiliza un valor por defecto para 'z'
  def initialize(x, y, z: 0)
    @x = restric_to_range(x)
    @y = restric_to_range(y)
    @z = restric_to_range(z)
  end
end
```

Gestión de Memoria

Java y Ruby: Gestión Automática

Aspecto	Descripción

Ubicación	Todos los objetos se crean en la **memoria dinámica (heap)**
Liberación	Automática mediante **Recolector de Basura (Garbage Collector)**
Condición	Cuando un objeto ya no está referenciado (deja de ser útil)

C++ (Contraste): Gestión Manual

 Diferencia: El programador decide si crea objetos en la **pila (stack)** o en el **heap**, y es responsable de la liberación manual usando el **Destructor** (~Clase()) y delete.

📊 Tabla Comparativa

Aspecto	Java	Ruby	C++
Nombre	Mismo que la clase	initialize	Mismo que la clase
Sobrecarga	Sí	No (usar alternativas)	Sí
Llamada	new ClassName()	ClassName.new	new ClassName()
Memoria	Automática (GC)	Automática (GC)	Manual
Reutilización	this()	Métodos de clase	Lista de inicialización

---

🎯 Buenas Prácticas

Práctica	Descripción	Prioridad
Inicializar TODOS los atributos	Evitar estados indefinidos	Crítico
Validación en constructor	Verificar restricciones del dominio	Crítico
Reutilizar código	Usar this() en Java o métodos auxiliares	Recomendado
Constructor por defecto	Proporcionar valores sensatos	Opcional

---

Adam Bourbahh Romero ~ PDOO