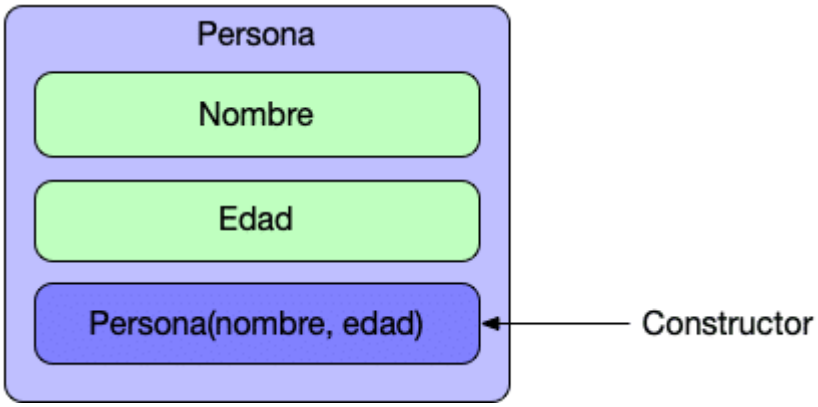
Tabla de Contenidos

El concepto de Java Constructor es uno de los conceptos más clásicos de programación orientada a objeto cuando nosotros diseñamos una clase para luego generar objetos de ella necesitamos una función constructora que inicialice las diferentes propiedades de esta clase. Veamos uno de los ejemplos más básicos con la clase Persona:

```
package com.arquitecturajava.ejemplo1;

public class Persona {

  private String nombre;
  private int edad;
  public Persona(String nombre, int edad) {

    this.nombre = nombre;
    this.edad = edad;
  }
  public String getNombre() {
    return nombre;
  }
  public void setNombre(String nombre) {
    this.nombre = nombre;
  }
  public int getEdad() {
    return edad;
  }
```

```
  public void setEdad(int edad) {
    this.edad = edad;
  }
}
```

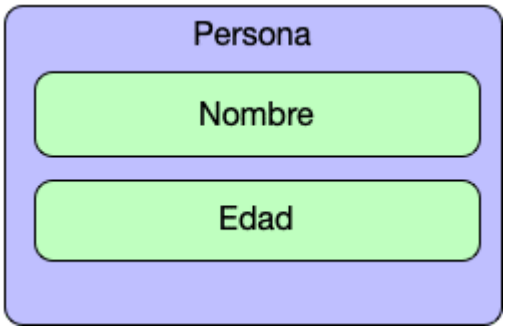
En Java para definir un Java Constructor debemos generar una función con el mismo nombre de la clase y pasarla los parámetros que consideremos adecuados a la hora de inicializar los objetos.



Hasta aquí todo correcto es momento de utilizarla:

```
package com.arquitecturajava.ejemplo1;

public class Principal {
        public static void main(String[] args) {
                Persona p= new Persona("juan",20);
        }
}
```

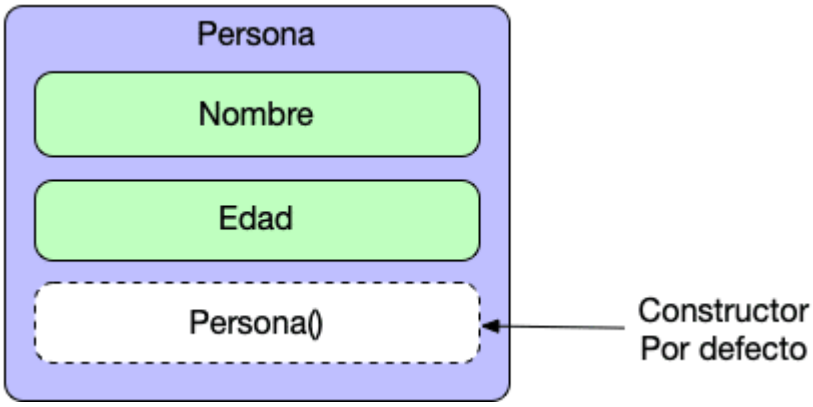
Hay muchas veces que nosotros no declaramos ningún constructor es decir definimos la clases sin constructores.



¿Que ocurre en este caso?

Constructores por defecto

La realidad es que toda clase necesita un constructor para poder instanciar los diferentes objetos . Por lo tanto el lenguaje de programación genera lo que se denomina un constructor por defecto que no recibe ningún parámetro.



Esto facilita la instanciación de objetos . Eso si este constructor solo será incluido por el compilador en el caso de que nuestra clase no disponga de ningún constructor.

```
package com.arquitecturajava.ejemplo2;

public class Persona {

        private String nombre;
        private int edad;
```

```
        public String getNombre() {
                return nombre;
        }
        public void setNombre(String nombre) {
                this.nombre = nombre;
        }
        public int getEdad() {
                return edad;
        }
        public void setEdad(int edad) {
                this.edad = edad;
        }
}
```

En una clase definida como sin constructor siempre podremos instanciar un objeto apoyándonos en el constructor por defecto que se añade a nivel de compilación:

```
package com.arquitecturajava.ejemplo2;

public class Principal {

        public static void main(String[] args) {
                Persona p= new Persona();

        }

}
```

Java Constructor y curiosidades

Si la clase dispone de un constructor el compilador no añadirá ningún constructor por defecto . Esto muchas veces a los programadores junior les choca pero es lógico ya que no

todas las posibilidades son admitidas a la hora de construir un objeto . Por lo tanto si como programadores decidimos añadir un constructor determinado Java supondrá que es el constructor adecuado y no añadirá el constructor por defecto por lo tanto con una clase persona como la siguiente :

```
package com.arquitecturajava.ejemplo1;

public class Persona {

	private String nombre;
	private int edad;
	public Persona(String nombre, int edad) {
		this.nombre = nombre;
		this.edad = edad;
	}
	public String getNombre() {
		return nombre;
	}
	public void setNombre(String nombre) {
		this.nombre = nombre;
	}
	public int getEdad() {
		return edad;
	}
	public void setEdad(int edad) {
		this.edad = edad;
	}
}
```

No se podrá generar un objeto utilizando un constructor por defecto o sin parámetros es decir este código no compilaría:

```
package com.arquitecturajava.ejemplo1;

public class Principal {

	public static void main(String[] args) {
		Persona p= new Persona();

	}

}
```

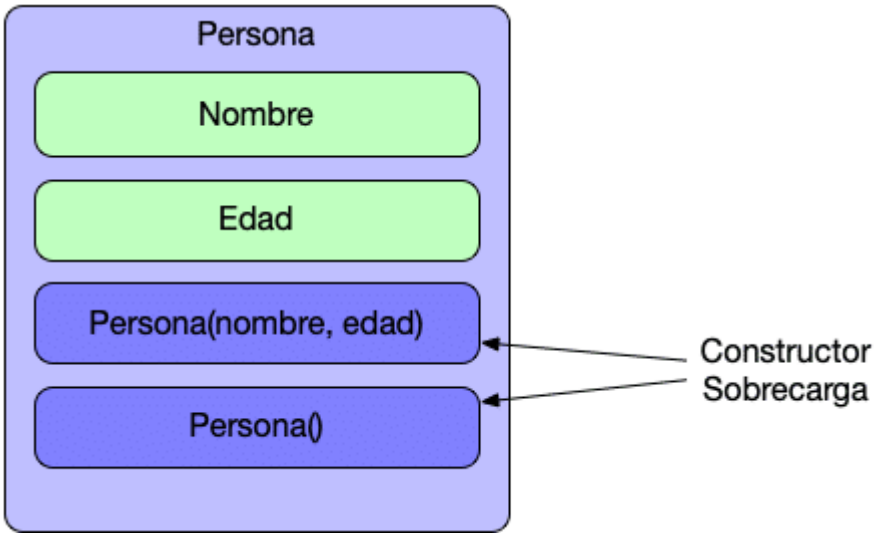
Java Constructor y Sobrecarga

Para conseguir que podamos tanto construir un objeto Persona pasando nombre y edad como construir un objeto Persona sin pasarle ningún parámetro deberemos hacer uso de la sobrecarga.

```
package com.arquitecturajava.ejemplo3;

public class Persona {

	private String nombre;
	private int edad;
	public String getNombre() {
		return nombre;
	}
	public void setNombre(String nombre) {
		this.nombre = nombre;
	}
	public int getEdad() {
		return edad;
	}
```

```
        public void setEdad(int edad) {
                this.edad = edad;
        }
        public Persona(String nombre, int edad) {
                super();
                this.nombre = nombre;
                this.edad = edad;
        }
        public Persona() {
        }
}
```

En este caso hemos definido dos constructores cada uno de los cuales cubre una casuística.



Permitiéndonos abordar ambas opciones sin problemas y construir un objeto Persona tanto pasando parámetros como sin pasarlos:

Otros artículos relacionados

- Java this vs this()
- Java Constructores this() y super()
- Java Override y encapsulación