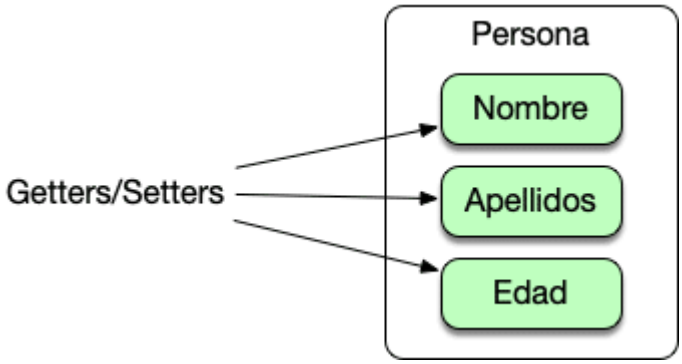
Tabla de Contenidos

- Propiedades privadas con métodos getters y Setters .
- Debe tener un constructor publico sin parámetros.
- Java Bean y Constructores por defecto
- Java Bean e Implementar Serializable
- Otros artículos relacionados

Un Java Bean es un estandar que hace referencia a la definición de clases de negocio con unos requisitos concretos . Vamos a ver cuales son estos requisitos que estamos obligados a cumplir.

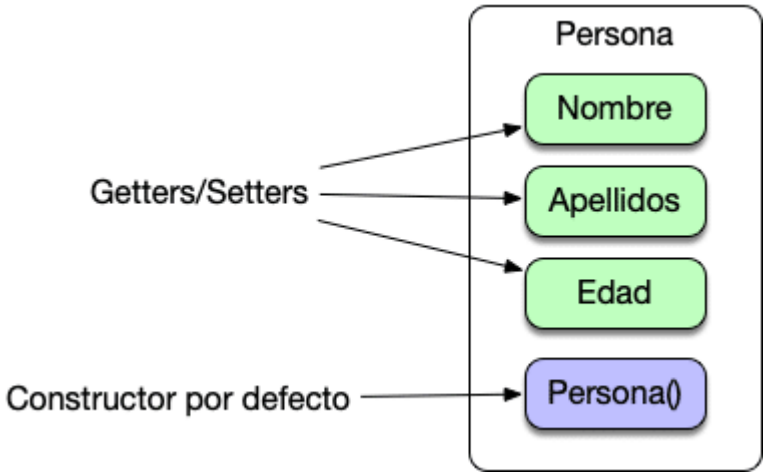
Propiedades privadas con métodos getters y Setters .

Esa es una de los primeros requerimientos todas las propiedades son privadas y se accede a través de Getters y Setters.



Debe tener un constructor publico sin parámetros.

Este es uno de los temas que más quebraderos de cabeza da a los desarrolladores a la hora de confirmar que algo es un Java Bean.



Si por ejemplo dispongo de la clase Persona y tiene un constructor como

```
public Persona (String ,nombre , String apellidos, int edad)  {}
```

Esto no lo convierte en JavaBean ya que no se trata de un constructor sin parámetros. Tampoco le convierte en Java Bean el siguiente :

```
Persona ()  {}
```

Ya que el constructor no es publico , para que sea un JavaBean debe disponer del siguiente constructor :

```
public Persona () {}
```

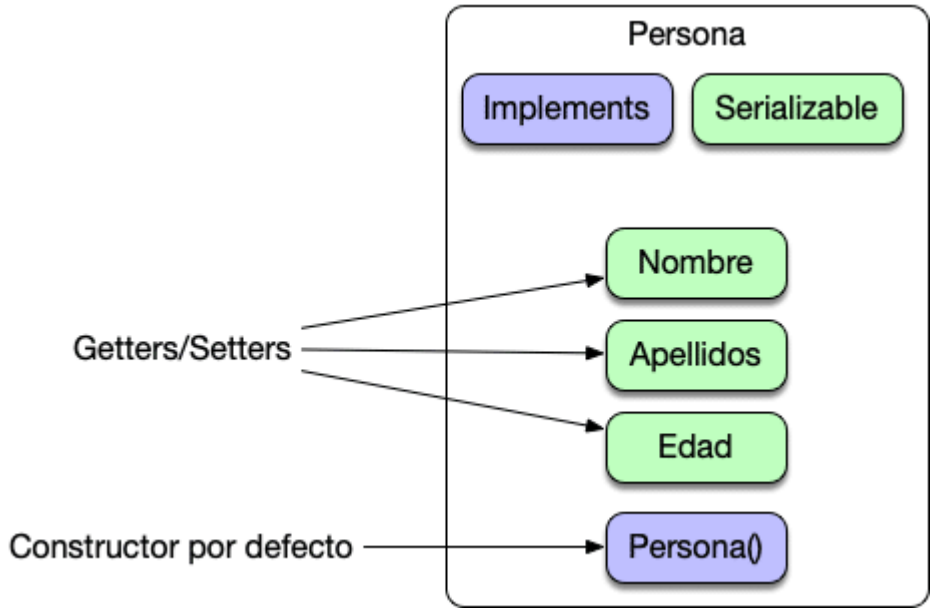
Eso automáticamente lo convierte en un JavaBean

Java Bean y Constructores por defecto

Tenemos que recordar que toda clase Java que no disponga de un constructor el compilador le añade un constructor por defecto . Por lo tanto si tenemos una clase sin constructor pero con los métodos Getter y Setter correctamente construidos si encajará como un JavaBean ¿Que más necesitamos para que la clase pueda ser considerada un Bean?

Java Bean e Implementar Serializable

Para que una clase se considere un Bean debe implementar el interface Serializable.



El interface Serializable es un interface de marca que no contiene ningún método pero que permite que los objetos sean serializables a disco o a red.

```
public class Personas implements Serializable
```

Recordemos que no hace falta que lo implemente ella misma sino que también lo puede implementar una clase Padre.

Por lo tanto resumiendo mucho la clase Persona es un JavaBean si es implementada de esta forma:

```
package com.arquitecturajava;

import java.io.Serializable;

public class Persona  implements Serializable{
```

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

Otros artículos relacionados

- Utilizando Spring MVC Bean Validation
- Cursos Arquitectura JAVA Fundae
- Spring @Bean y su uso
- ¿Que es un Java Predicate?