

# Lombok

# ¿Que es?

- ▶ Una librería cuyo objetivo es reducir la cantidad de “Boilerplate Code” en nuestro código fuente. Por ejemplo, getters, setters, constructores, etc.
- ▶ Trabaja a través de anotaciones que se agregan a nuestras clases.
- ▶ Trabaja en tiempo de compilación, no genera overhead durante la ejecución.
- ▶ Permite mantener un código fuente más limpio y legible.

# Algunas Anotaciones

- ▶ @Getter
- ▶ @Setter
- ▶ @NoArgsConstructor
- ▶ @AllArgsConstructor
- ▶ @RequiredArgsConstructor
- ▶ @Builder
- ▶ @ToString
- ▶ @EqualsAndHashCode
- ▶ @Data
- ▶ @Log
- ▶ @Slf4j

# @Data

- ▶ Se suele ver mucho y es una agrupación de otras anotaciones.
- ▶ Es equivalente a:
  - ▶ @Getter
  - ▶ @Setter
  - ▶ @RequiredArgsConstructor
  - ▶ @ToString
  - ▶ @EqualAndHashCode

# Ejemplo

```
@AllArgsConstructor  
@NoArgsConstructor  
@Getter  
@Setter  
@ToString  
public class UsuarioLombok {  
    private String nombre;  
    private String apellido;  
    private String email;  
    private String direccion;  
    private String telefono;  
    private Date fechaNacimiento;  
}
```

▼   **UsuarioLombok**

-   **UsuarioLombok(String, String, String, String, String, Date)**
-   **UsuarioLombok()**
-   **getNombre(): String**
-   **getApellido(): String**
-   **getEmail(): String**
-   **getDireccion(): String**
-   **getTelefono(): String**
-   **getFechaNacimiento(): Date**
-   **setNombre(String): void**
-   **setApellido(String): void**
-   **setEmail(String): void**
-   **setDireccion(String): void**
-   **setTelefono(String): void**
-   **setFechaNacimiento(Date): void**
-   **toString(): String** ↑ **Object**

# Ejemplo Builder

```
@Builder  
@Getter  
public class Menu {  
    String plato;  
    String postre;  
    String bebida;  
    String platoAlternativo;  
    String postreAlternativo;  
    String bebidaAlternativa;  
}
```

```
▼ © ⌂ Menu  
  > ⚡ ⌂ MenuBuilder  
    ⚡ ⌂ Menu(String, String, String, String, String, String)  
    ⚡ ⌂ getPlato(): String  
    ⚡ ⌂ getPostre(): String  
    ⚡ ⌂ getBebida(): String  
    ⚡ ⌂ getPlatoAlternativo(): String  
    ⚡ ⌂ getPostreAlternativo(): String  
    ⚡ ⌂ getBebidaAlternativa(): String  
    ⚡ ⌂ builder(): MenuBuilder
```

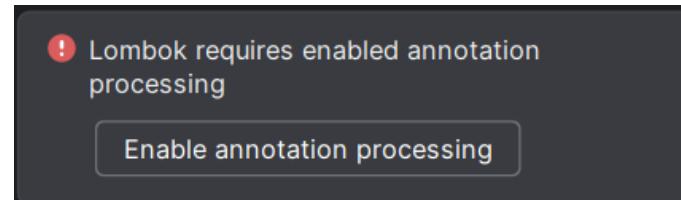
```
Menu menu = Menu.builder()  
    .plato("Hamburguesa")  
    .postre("Helado")  
    .build();
```

# ¿Como instalarlo con Maven?

- ▶ Agregamos a nuestro pom la siguiente dependencia

```
<dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <version>1.18.34</version>
    <scope>provided</scope>
</dependency>
```

- ▶ Segundo nuestro IDE por ahí nos pide activar el procesamiento de anotaciones



# Diferencias entre generar el código con ayuda de nuestro IDE y usando Lombok

- ▶ Si bien el código resultante sea lo mismo, con Lombok la clase en el código fuente quedara mucho más limpia

Con Lombok

```
@AllArgsConstructor  
@NoArgsConstructor  
@Getter  
@Setter  
@ToString  
  
public class UsuarioLombok {  
    private String nombre;  
    private String apellido;  
    private String email;  
    private String direccion;  
    private String telefono;  
    private Date fechaNacimiento;  
}
```

Sin Lombok

```
public class Usuario {  
    private String nombre;  
    private String apellido;  
    private String email;  
    private String direccion;  
    private String telefono;  
    private Date fechaNacimiento;  
    private boolean habilitado;  
  
    public Usuario() {}  
  
    public Usuario(String nombre, String apellido, String email, String direccion, String telefono, Date fechaNacimiento, boolean habilitado) {  
        this.nombre = nombre;  
        this.apellido = apellido;  
        this.email = email;  
        this.direccion = direccion;  
        this.telefono = telefono;  
        this.fechaNacimiento = fechaNacimiento;  
        this.habilitado = habilitado;  
    }  
  
    public boolean isHabilitado() {  
        return habilitado;  
    }  
  
    public void setHabilitado(boolean habilitado) {  
        this.habilitado = habilitado;  
    }  
  
    public String getNombre() {  
        return nombre;  
    }  
  
    public void setNombre(String nombre) {  
        this.nombre = nombre;  
    }  
  
    public String getApellido() {  
        return apellido;  
    }  
  
    public void setApellido(String apellido) {  
        this.apellido = apellido;  
    }  
  
    public String getEmail() {  
        return email;  
    }  
  
    public void setEmail(String email) {  
        this.email = email;  
    }  
  
    public String getDireccion() {  
        return direccion;  
    }  
  
    public void setDireccion(String direccion) {  
        this.direccion = direccion;  
    }  
  
    public String getTelefono() {  
        return telefono;  
    }  
  
    public void setTelefono(String telefono) {  
        this.telefono = telefono;  
    }  
  
    public Date getFechaNacimiento() {  
        return fechaNacimiento;  
    }  
  
    @Override  
    public String toString() {  
        return "Usuario{" +  
            "nombre='" + nombre + '\'' +  
            ", apellido='" + apellido + '\'' +  
            ", email='" + email + '\'' +  
            ", direccion='" + direccion + '\'' +  
            ", telefono='" + telefono + '\'' +  
            ", fechaNacimiento=" + fechaNacimiento +  
            ", habilitado=" + habilitado +  
            '}';  
    }  
}
```

# Bibliografía

- ▶ Página Oficial <https://projectlombok.org/>
- ▶ Documentación <https://projectlombok.org/features/>
- ▶ Instalación <https://projectlombok.org/setup/maven>
- ▶ Repositorio de Ejemplos <https://github.com/tomasdozo/ttps>

# Anexo Logs

- ▶ A nivel profesional, en lugar de imprimir los mensajes en consola con System.out para saber que pasa en el sistema, se utiliza lo que se conoce como Logger, existen muchas librerías e implementaciones.
- ▶ Son altamente customizables.
- ▶ Su uso básico es muy sencillo.

Ejemplos usando Logback y la anotacion @Slf4j de Lombok.  
Los metodos y salidas puede variar con otras implementaciones.

Codigo

```
log.info("Se creo el usuario: {}", usuario);
log.info("Se creo un menu");
log.warn("No tiene menu alternativo");
log.error("Falta bebida");
```

Consola

```
20:59:07.794 [main] INFO ttps.lombok.ServicioBuffet -- Se creo el usuario: UsuarioLombok(nombre=F
20:59:07.807 [main] INFO ttps.lombok.ServicioBuffet -- Se creo un menu
20:59:07.807 [main] WARN ttps.lombok.ServicioBuffet -- No tiene menu alternativo
20:59:07.807 [main] ERROR ttps.lombok.ServicioBuffet -- Falta bebida
```