

Index Controlador con Spring Boot



📌 Guía: Creación del Controlador `IndexControlador.java` en Spring Boot con JSF

1 Introducción

En esta sección, crearemos el **controlador `IndexControlador.java`**, que se encargará de manejar las solicitudes de la vista **JSF** y conectar con la capa de servicio para obtener los datos de los clientes.

📌 ¿Qué aprenderemos?

- Crear un **controlador gestionado por Spring Boot**.
- Implementar **inyección de dependencias** con `@Autowired`.
- Usar `@ViewScoped` para gestionar el ciclo de vida de la vista.
- Integrar **Lombok (`@Data`)** para reducir el código repetitivo.
- Usar `@PostConstruct` para inicializar datos al cargar la vista.

2 Creación de la Clase `IndexControlador.java`

📌 Ubicación del archivo en el proyecto: src/main/java/gm/zona_fit/controlador/IndexControlador.java**💡 Código actualizado de IndexControlador.java**

```
package gm.zona_fit.controlador;

import gm.zona_fit.modelo.Cliente;
import gm.zona_fit.servicio.IClienteServicio;
import jakarta.annotation.PostConstruct;
import jakarta.faces.view.ViewScoped;
import lombok.Data;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;
import java.util.List;
import org.slf4j.Logger;

@Component
@Data
@ViewScoped
public class IndexControlador {

    @Autowired
    IClienteServicio clienteServicio;
    private List<Cliente> clientes;
    private static final Logger logger = LoggerFactory.getLogger(IndexControlador.class);

    @PostConstruct
    public void init(){
        cargarDatos();
    }

    public void cargarDatos(){
        this.clientes = this.clienteServicio.listarClientes();
        this.clientes.forEach(cliente -> logger.info(cliente.toString()));
    }
}
```

3 Explicación del Código**📌 1. Definición de la clase y anotaciones principales**

```
@Data
@Component
```

```
@ViewScope  
@Slf4j
```

- @Data (Lombok)** → Genera automáticamente los métodos **getters/setters**, **toString**, y **equals/hashCode**.
 - @Component** → Indica que esta clase es un **componente de Spring**, permitiendo que sea gestionada por la fábrica de beans de Spring Boot.
 - @ViewScope** → Mantiene los datos mientras la vista esté cargada (útil para aplicaciones **de una sola página con JSF**).
-

📌 2. Inyección del Servicio `IClienteServicio`

```
@Autowired  
private IClienteServicio clienteServicio;
```

- @Autowired** → Inyecta la dependencia del servicio que maneja los datos de los clientes.
-

📌 3. Lista de Clientes

```
private List<Cliente> clientes;
```

- Variable que almacena la lista de clientes recuperados desde la base de datos.
-

📌 4. Método `init()` para inicializar datos

```
@PostConstruct  
public void init() {  
    cargarDatos();  
}
```

- @PostConstruct** → Se ejecuta después de que Spring crea la instancia de la clase.
 - Llama a `cargarDatos()`** para recuperar los clientes de la base de datos al cargar la vista.
-

📌 5. Método `cargarDatos()` para obtener clientes

```
public void cargarDatos() {  
    clientes = clienteServicio.listarClientes();  
    clientes.forEach(cliente -> log.info(cliente.toString()));  
}
```

- Obtiene los clientes usando el servicio `IClienteServicio`.
 - Registra cada cliente en la **bitácora del sistema** (`log.info(...)`).
-

Conclusión

De momento aún no vamos a ejecutar nuestra aplicación. Esto lo haremos en la siguiente lección.

Con la implementación de `IndexControlador.java`, ahora nuestra aplicación:

- Carga automáticamente la lista de clientes** al iniciar la vista.
- Usa Spring Boot y Lombok** para simplificar el código.
- Mantiene el estado de los datos** mientras la vista esté activa.
- Registra la información en el log** para depuración.

 ¡Listo para continuar con la integración de la vista en JSF!

Nota: Recuerda que tienes el archivo .zip en la sección de recursos de este video con todos los archivos del proyecto actualizado según la lección estudiada, el cual puedes utilizar para cualquier problema.

Sigue adelante con tu aprendizaje  , ¡el esfuerzo vale la pena!

¡Saludos! 

Ing. Ubaldo Acosta

Fundador de GlobalMentoring.com.mx