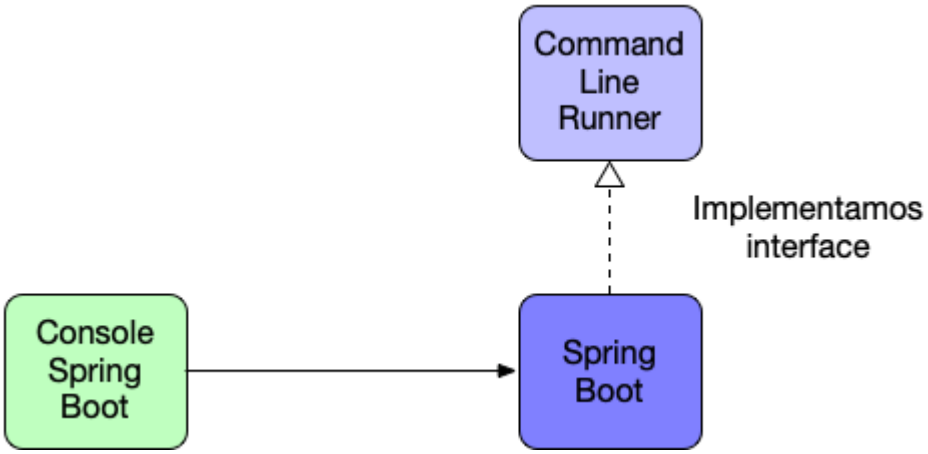
Spring Boot Console , es una de las posibilidades que tenemos con Spring Boot . Podemos construir un proyecto que se ejecute desde la consola pura. No tenemos porque disponer de un aplicativo web clásico. Spring Boot nos permite arrancar una aplicación no importa mucho su tipologia. Para conseguir que esto sea operativo .



Hay pocas cosas que tenemos que hacer, es muy sencillo . Como siempre arrancaremos el proyecto de Boot incluyendo las dependencias necesarias. En este caso no tendremos dependencias web.

```
<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-
jdbc</artifactId>
		</dependency>
		<dependency>
			<groupId>com.mysql</groupId>
			<artifactId>mysql-connector-j</artifactId>
			<scope>runtime</scope>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-
```

```
test</artifactId>
                        <scope>test</scope>
                </dependency>
        </dependencies>
```

En este caso hemos optado por MySQL y Spring Data JDBC. No necesitamos más para tener una aplicación que se conecte a una base de datos. Diseñamos un repositorio muy sencillo con anotaciones

```
import org.springframework.transaction.annotation.Transactional;

@Repository
public class PersonaRepository {
  private JdbcTemplate plantilla;
  public PersonaRepository(JdbcTemplate plantilla) {
    this.plantilla = plantilla;
  }
  @Transactional
  public void insertar(Persona persona) {
  plantilla.update("insert into Personas values (?,?,?)",
persona.getNombre(), persona.getApellidos(),
        persona.getEdad());
  }
}
```

Spring Boot Console y compomentes

En este caso hemos usado JDBCTemplates para acceder a la base de datos . De igual manera construimos una clase de Servicio que llame al repositorio.

```
package com.arquitecturajava.aop;
```

```
public class PersonaService {
	private PersonaRepository repositorio;

	public void insertar(Persona persona) {
		repositorio.insertar(persona);
	}
}
```

Nos queda diseñar la clase Persona como clase de negocio:

```
package com.arquitecturajava.aop;

public class Persona {

	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(String nombre) {
		this.nombre = nombre;
	}

}
```

El interface CommandLineRunner

Modificamos el fichero de Aplicación de Spring Boot y le obligamos a que implemente el interface CommandLineRunner que será encargado de ejecutar los servicios y repositorios de la aplicación los cuales hemos inyectado. Acabamos de construir un Spring Boot Console Application

```
package com.arquitecturajava.aop;
```

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class AopApplication implements CommandLineRunner {
	@Autowired
	private PersonaRepository repositorio;
	public static void main(String[] args) {
		SpringApplication.run(AopApplication.class, args);
	}

	@Override
	public void run(String... args) throws Exception {
		Persona p= new Persona("2","ana",20);
		repositorio.insertar(p);
	}

}
```

Como se puede observar la clases añade un método run e inyecta con las anotaciones de Spring las dependencias necesarias. Para luego ejecutar el programa e insertar registros en la base de datos. Así de sencillo es trabajar con Spring Boot Console y aplicaciones

- Introducción a Spring Data y JPA
- Curso Spring Boot y MicroServicios
- ¿Qué es Spring Boot?
- Spring Boot JPA y su configuración
- Spring Boot JSP y su configuración
- Spring Boot