Cómo usar Spring Boot CLI

Spring Boot CLI (Interfaz de Línea de Comandos) es una herramienta poderosa que te permite desarrollar, probar y prototipar aplicaciones Spring Boot directamente desde la línea de comandos. Es especialmente útil para ejecutar scripts de Groovy sin configurar una estructura de proyecto completa, generar nuevos proyectos o experimentar con las características de Spring Boot. A continuación, se presenta una guía completa sobre cómo instalar y usar Spring Boot CLI de manera efectiva.

Instalación

Antes de poder usar Spring Boot CLI, necesitas instalarlo. Hay dos métodos principales dependiendo de tu sistema operativo:

1. Usando SDKMAN! (Recomendado para Sistemas Basados en Unix como Linux o macOS)

SDKMAN! es una herramienta para gestionar kits de desarrollo de software, lo que la convierte en una forma fácil de instalar Spring Boot CLI.

- **Paso 1: Instalar SDKMAN!** Abre tu terminal y ejecuta:

  ```
  curl -s "https://get.sdkman.io" | bash
  ```

  Sigue las instrucciones para inicializar SDKMAN! al ejecutar el script:

  ```
  source "$HOME/.sdkman/bin/sdkman-init.sh"
  ```

- **Paso 2: Instalar Spring Boot CLI** Ejecuta el siguiente comando:

  ```
  sdk install springboot
  ```

2. Instalación Manual (Para Windows o Configuración Manual)

Si estás en Windows o prefieres una instalación manual: - Descarga el archivo ZIP de Spring Boot CLI desde el sitio web oficial de Spring. - Extrae el archivo ZIP a un directorio de tu elección. - Añade el directorio `bin` de la carpeta extraída a la variable de entorno PATH de tu sistema.

Verificar la Instalación

Para confirmar que Spring Boot CLI se ha instalado correctamente, ejecuta este comando en tu terminal:

```
spring --version
```

Deberías ver la versión instalada de Spring Boot CLI (por ejemplo, `Spring CLI v3.3.0`). Si esto funciona, ¡estás listo para empezar a usarlo!

Formas Clave de Usar Spring Boot CLI

Spring Boot CLI proporciona varias características que lo hacen ideal para el desarrollo y prototipado rápidos. Aquí están las formas principales de usarlo:

1. Ejecutar Scripts de Groovy

Una de las características destacadas de Spring Boot CLI es su capacidad para ejecutar scripts de Groovy directamente sin requerir una configuración de proyecto completa. Esto es perfecto para el prototipado rápido o la experimentación con Spring Boot.

- **Ejemplo: Crear una Aplicación Web Simple** Crea un archivo llamado `hello.groovy` con el siguiente contenido:

```
@RestController
class HelloController {
    @RequestMapping("/")
    String home() {
        "¡Hola, Mundo!"
    }
}
```

- **Ejecutar el Script** En tu terminal, navega al directorio que contiene `hello.groovy` y ejecuta:

```
spring run hello.groovy
```

  Esto inicia un servidor web en el puerto 8080. Abre un navegador y visita `http://localhost:8080` para ver "¡Hola, Mundo!"mostrado.

- **Añadir Dependencias** Puedes incluir dependencias directamente en el script usando la anotación `@Grab`. Por ejemplo:

```
@Grab('org.springframework.boot:spring-boot-starter-data-jpa')
@RestController
class HelloController {
    @RequestMapping("/")
    String home() {
        "¡Hola, Mundo!"
```

```
    }
}
```

Esto añade Spring Data JPA a tu script sin necesidad de un archivo de construcción.

- **Ejecutar Múltiples Scripts** Para ejecutar todos los scripts de Groovy en el directorio actual, usa:

```
spring run *.groovy
```

2. Crear Nuevos Proyectos Spring Boot

Spring Boot CLI puede generar una nueva estructura de proyecto con las dependencias deseadas, ahorrándote tiempo al comenzar una aplicación completa.

- **Ejemplo: Generar un Proyecto** Ejecuta este comando para crear un nuevo proyecto con dependencias web y data-jpa:

```
spring init --dependencies=web,data-jpa my-project
```

Esto crea un directorio llamado `my-project` que contiene una aplicación Spring Boot configurada con Spring Web y Spring Data JPA.

- **Opciones de Personalización** Puedes especificar opciones adicionales como:
  - Herramienta de construcción: `--build=maven` o `--build=gradle`
  - Lenguaje: `--language=java`, `--language=groovy`, o `--language=kotlin`
  - Empaquetado: `--packaging=jar` o `--packaging=war`

  Por ejemplo:

```
spring init --dependencies=web --build=gradle --language=kotlin my-kotlin-project
```

3. Empaquetar Aplicaciones

Spring Boot CLI te permite empaquetar tus scripts en archivos JAR o WAR ejecutables para su despliegue.

- **Crear un Archivo JAR**

```
spring jar my-app.jar *.groovy
```

Esto empaqueta todos los scripts de Groovy en el directorio actual en `my-app.jar`.

- **Crear un Archivo WAR**

```
spring war my-app.war *.groovy
```

Esto genera un archivo `my-app.war` adecuado para su despliegue en un contenedor de servlets.

4. Ejecutar Pruebas

Si tienes scripts de prueba de Groovy, puedes ejecutarlos con:

```
spring test *.groovy
```

Esto ejecuta todos los scripts de prueba en el directorio actual.

5. Usar la Consola Interactiva

Para una experiencia interactiva, lanza la consola de Spring Boot CLI:

```
spring shell
```

Dentro de la consola, puedes ejecutar comandos como `run`, `grab` o `jar` de manera interactiva, lo cual es genial para la experimentación.

Resumen

Spring Boot CLI es una excelente herramienta para desarrolladores que desean trabajar con Spring Boot rápidamente y de manera eficiente desde la línea de comandos. Aquí está cómo usarlo en resumen:

1. **Instálalo** usando SDKMAN! (`sdk install springboot`) o manualmente descargando el ZIP y actualizando tu PATH.
2. **Ejecuta scripts de Groovy** con `spring run <script.groovy>` para el prototipado rápido.
3. **Crea nuevos proyectos** con `spring init --dependencies=<deps> <project-name>`.
4. **Empaqueta aplicaciones** en archivos JAR o WAR usando `spring jar` o `spring war`.
5. **Experimenta de manera interactiva** con `spring shell`.

Ya sea que estés prototipando una idea, aprendiendo Spring Boot o automatizando tareas, Spring Boot CLI ofrece una manera ligera y flexible de comenzar sin el sobrecoste de un IDE completo o una configuración de proyecto.