Aquí tienes el texto mejorado:

El uso de propiedades en Spring Boot es muy común cuando trabajamos con una aplicación de Spring Boot . A diferencia de las aplicaciones clásicas del Spring Framework, Spring Boot aplica el principio de convención sobre configuración y define un archivo de propiedades por defecto. Este archivo se encuentra en la carpeta `resources` de nuestro proyecto.

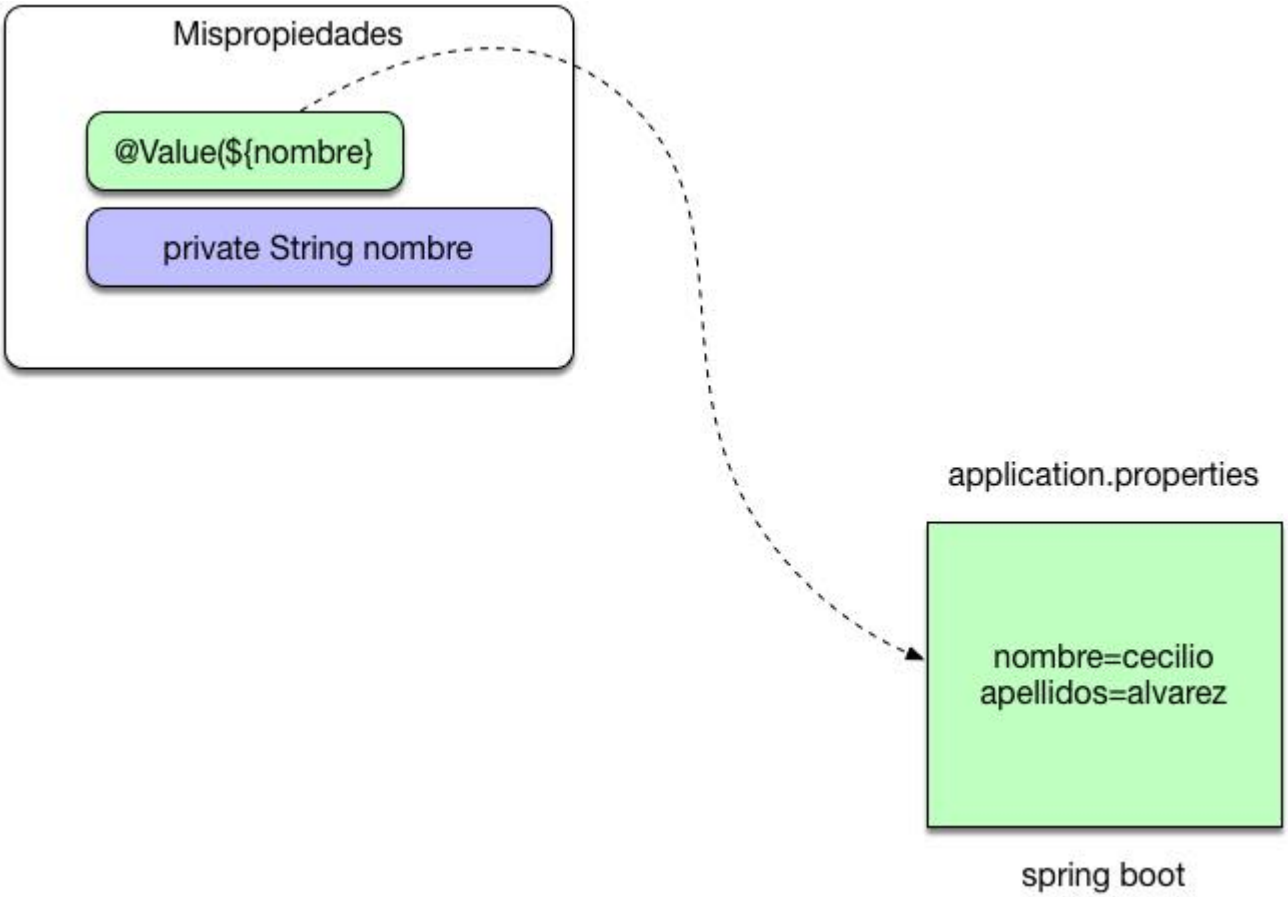
Spring Boot Properties

Vamos a utilizar este archivo de propiedades para registrar dos valores (un nombre y un apellido) y acceder a ellos desde nuestra aplicación. Debemos configurar nuestra aplicación Spring Boot con las anotaciones necesarias. En nuestro caso, definiremos una clase Java sencilla que contenga ambas propiedades.

```
package com.arquitecturajava.springPrueba;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component
public class MisPropiedades {

    @Value("${nombre}")
    private String nombre;
```

```
    @Value("${apellidos}")
    private String apellidos;

    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;
    }

}
```

Como podemos observar nos hemos apoyado en la anotación @Value para inicializar las propiedades.



Hemos hecho uso de spring expression language para acceder a cada uno de los valores utilizando ${valor} . De esta forma tan sencilla seremos ya capaces de acceder a las propiedades definidas en el fichero application.properties. Nos queda simplemente inyectar nuestro clase en el proyecto de SpringBoot y ejecutarlo como proyecto de consola para ver el resultado.

```
package com.arquitecturajava.springPrueba;

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 SpringPruebaApplication implements CommandLineRunner {

    @Autowired
    private MisPropiedades misPropiedades;

    public static void main(String[] args) {
        SpringApplication app = new
SpringApplication(SpringPruebaApplication.class);
        app.run(args);
    }

    @Override
    public void run(String... args) throws Exception {
        System.out.println(misPropiedades.getNombre());
        System.out.println(misPropiedades.getApellidos());
    }
}
```

Ahora nos es suficiente con ejecutar nuestra aplicación desde Eclipse y veremos como se imprime el nombre y los apellidos por la consola.

```
2017-11-29 10:46:42.101  INFO 40772 --- [
2017-11-29 10:46:42.104  INFO 40772 --- [
2017-11-29 10:46:42.167  INFO 40772 --- [
2017-11-29 10:46:42.777  INFO 40772 --- [
cecilio
alvarez
2017-11-29 10:46:42.798  INFO 40772 --- [
2017-11-29 10:46:42.799  INFO 40772 --- [
2017-11-29 10:46:42.800  INFO 40772 --- [
```

Acabamos de utilizar las capacidades de Spring Boot para manejo de propiedades.

Otros artículos relacionados

1. Spring Boot WAR sin Microservicios
2. ¿Qué es Spring Boot?
3. Spring Boot AOP y rendimiento
4. Spring Boot