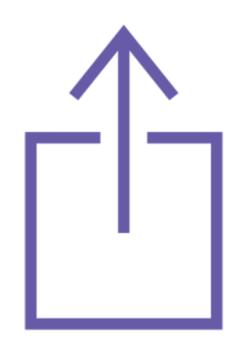
Working with Spring Boot Config and Environment Needs



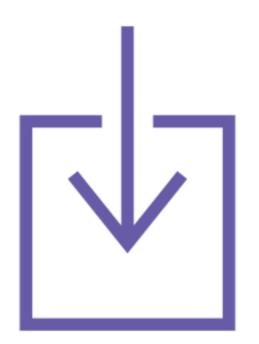
Dan Bunker
CONSULTANT | SAAS BUILDER | MENTOR
linkedin.com/in/bunkerdan

How to Customize and Override Spring Boot



External Sources

Command line parameters, JNDI, OS environment variables



Internal Sources

Servlet parameters, property files, java configuration

How to Customize and Override Spring Boot



Order of precedence:

- 1. Command line args
- 2. SPRING_APPLICATION_JSON args
- 3. Servlet parameters
- 4. JNDI
- 5. Java System Properties
- 6. OS environment variable
- 7. Profile properties
- 8. Application properties
- 9. @PropertySource annotations
- 10. Default properties

Clip 3 - Overview Placeholder

Demo: Locating and Working with Config Files

Clip 4 - Overview Placeholder

Demo: How to Change Database Connections

Demo: How to Handle Different Environments

Environmental Configuration

application-{profile}.properties

application-dev.properties

application-prod.properties

-Dspring.profiles.active=dev

Clip 6 - Overview Placeholder

Demo: Setting Properties with YAML

Clip 7 - Overview Placeholder

Demo: How to Use Spring Properties in Code

Clip 8 - Overview Placeholder

Demo: Overriding Spring Boot with Java Config

JAR

```
@Configuration
public class MyAutoConfiguration {
    @Bean
    public SomeObject myMethod(){
        ...
    }
}
```

resources/META-INF/spring.factories

org.springframework.boot.autoconfigure.EnableAutoConfiguration=\
com.mypackage.MyAutoConfiguration

- @ConditionalOnClass
- @ConditionalOnMissingClass
- @ConditionalOnBean
- @ConditionalOnMissingBean
- @ConditionalOnProperty
- @ConditionalOnResource

Summary

Location, types and precedence

- src/main/resources
- properties and ymls

Default/common Spring Boot properties

Environment specific properties

Java configuration and autoconfiguration