Spring Boot

CS544: Enterprise Architecture

Spring Boot

CS544 Enterprise Architecture

Wholeness

- Spring Boot tries to get your project setup as quick as possible.
- It takes “an opinionated view of the Spring platform and third party libraries so you can get started with minimal fuss”.
- Do Less and Accomplish more.

Spring Boot:

DEPENDENCY MANAGEMENT

Typical pom.xml: Starter Parent

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
        <modelVersion>4.0.0</modelVersion>

        <groupId>com.example</groupId>
        <artifactId>myproject</artifactId>
        <version>0.0.1-SNAPSHOT</version>

        <!-- Inherit defaults from Spring Boot -->
        <parent>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-parent</artifactId>
                <version>2.0.4.RELEASE</version>
        </parent>

        <!-- Add typical dependencies for a web application -->
        <dependencies>
                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-web</artifactId>
                </dependency>
        </dependencies>

        <!-- Package as an executable jar -->
        <build>
                <plugins>
                        <plugin>
                                <groupId>org.springframework.boot</groupId>
                                <artifactId>spring-boot-maven-plugin</artifactId>
                        </plugin>
                </plugins>
        </build>

</project>
```

(Almost) all other Dependency version numbers set by Spring Boot

Plugin packages it as an executable JAR

Many Spring Boot Starters

So that you manage less individual dependencies

Table 13.1. Spring Boot application starters

Name	Description	
`spring-boot-starter`	Core starter, including auto-configuration support, logging and YAML	
`spring-boot-starter-activemq`	Starter for JMS messaging using Apache ActiveMQ	
`spring-boot-starter-amqp`	Starter for using Spring AMQP and Rabbit MQ	
`spring-boot-starter-aop`	Starter for aspect-oriented programming with Spring AOP and AspectJ	Pom
`spring-boot-starter-artemis`	Starter for JMS messaging using Apache Artemis	Pom
`spring-boot-starter-batch`	Starter for using Spring Batch	Pom
`spring-boot-starter-cache`	Starter for using Spring Framework's caching support	Pom
`spring-boot-starter-cloud-connectors`	Starter for using Spring Cloud Connectors which simplifies connecting to services in cloud platforms like Cloud Foundry and Heroku	Pom
`spring-boot-starter-data-cassandra`	Starter for using Cassandra distributed database and Spring Data Cassandra	Pom
`spring-boot-starter-data-cassandra-reactive`	Starter for using Cassandra distributed database and Spring Data Cassandra Reactive	Pom
`spring-boot-starter-data-couchbase`	Starter for using Couchbase document-oriented database and Spring Data Couchbase	Pom
`spring-boot-starter-data-couchbase-reactive`	Starter for using Couchbase document-oriented database and Spring Data Couchbase Reactive	Pom
`spring-boot-starter-data-elasticsearch`	Starter for using Elasticsearch search and analytics engine and Spring Data Elasticsearch	Pom
`spring-boot-starter-data-jpa`	Starter for using Spring Data JPA with Hibernate	Pom
`spring-boot-starter-data-ldap`	Starter for using Spring Data LDAP	Pom
`spring-boot-starter-data-mongodb`	Starter for using MongoDB document-oriented database and Spring Data MongoDB	Pom
`spring-boot-starter-data-mongodb-reactive`	Starter for using MongoDB document-oriented database and Spring Data MongoDB Reactive	Pom
`spring-boot-starter-data-neo4j`	Starter for using Neo4j graph database and Spring Data Neo4j	Pom
`spring-boot-starter-data-redis`	Starter for using Redis key-value data store with Spring Data Redis and the Lettuce	Pom

https://docs.spring.io/spring-boot/docs/2.0.4.RELEASE/reference/htmlsingle/#using-boot-starter

Spring Boot

CODE & CONFIGURATION

Java Config / Main Class

- Spring Boot favors Java-based configuration. Although it is possible to use SpringApplication with XML sources, we generally recommend that your primary source be a single @Configuration class. Usually the class that defines the main method is a good candidate as the primary @Configuration.
- Many Spring configuration examples have been published on the Internet that use XML configuration. If possible, always try to use the equivalent Java-based configuration. Searching for Enable* annotations can be a good starting point.

https://docs.spring.io/spring-boot/docs/2.0.4.RELEASE/reference/htmlsingle/#using-boot-configuration-classes

From COV2

Application.java

Source History

```java
package cs544.cov1;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

@SpringBootApplication
@EnableWebSecurity
public class Application {

    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }

}
```

Auto Configuration

- @SpringBootApplication

- Same as writing the following 3:
  - @EnableAutoConfiguration
  - @ComponentScan
  - @Configuration

- Application.java therefore is a JavaConfig class

Properties or YML

- Autoconfiguration still needs certain values
  - Such as database username and password

- These can be stored in:

  application.properties

  Or application.yml (YAML is a superset of JSON)

application.properties

Source History

```
spring.datasource.url = jdbc:mysql://localhost/cs544
spring.datasource.username = root
spring.datasource.password = root

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5Dialect
spring.jpa.hibernate.ddl-auto = create-drop

spring.mvc.view.prefix=/WEB-INF/views/
spring.mvc.view.suffix=.jsp

logging.level.root=DEBUG
```

Additional Configuration

- **Importing Additional Configuration Classes**
  - You need not put all your @Configuration into a single class. The @Import annotation can be used to import additional configuration classes. Alternatively, you can use @ComponentScan to automatically pick up all Spring components, including @Configuration classes.
- **Importing XML Configuration**
  - If you absolutely must use XML based configuration, we recommend that you still start with a @Configuration class. You can then use an @ImportResource annotation to load XML configuration files.

https://docs.spring.io/spring-boot/docs/2.0.4.RELEASE/reference/htmlsingle/#using-boot-configuration-classes

Spring Boot

RUNNING AN APPLICATION

Running

- For our purposes you just run it from your IDE
  - Can also run it from its own Maven target
  - Or run the executable JAR it creates

- You can also include devtools
  - Automatically disabled when running the JAR

```
<dependencies>
        <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-devtools</artifactId>
                <optional>true</optional>
        </dependency>
</dependencies>
```

DevTools Benefits

- Automatically restarts when it senses changes
- On restart logs changes to autoconfig (delta)
- Option to configure resources that should not trigger a restart when changed
- Also the ability to watch extra resources / paths for changes to trigger restart

Active Learning

- What 3 things does @SpringApplication do?

- What would you say is the main benefit of the Spring Boot Dev Tools?

Main Point

- Spring Boot tries to setup as much as possible for you.
- Science of Consciousness: take the right angle and let go.