Java → Spring Boot → Getting started with Spring Boot

# Theory: Getting started with Spring Boot

⏱ 4 minutes    0 / 2 problems solved

[ Skip this topic ]    [ Start practicing ]

In this topic, you will learn how to create your first Spring Boot application. This application is quite simple and does nothing useful, but still, it will demonstrate the basic structure of any Spring Boot project.

## §1. Generating a Spring Boot application

It is often hard to start a new project from scratch, especially for beginners. You need to set up the basic structure, create configurations and manage external dependencies. Fortunately, there is a special web site called Spring Initializr that can generate the basic Spring Boot project structure based on your wishes.



This site may look slightly different, but the general idea remains the same. There are several important options you need to select before generating a project:

- build tool: Maven or Gradle
- one of the possible JVM-based languages: Java, Kotlin, Groovy
- a version of Spring Boot framework
- metadata like `group`, `artifact`, `description` and `package name`
- external dependencies (other frameworks and libraries)

In our example, we've chosen Gradle as the build tool and Java 8 as the language. You can choose a more recent version. Our application is named `demo`.

By default, the packaging type for Spring Boot is **Jar.** It means that your application should be packed in a `.jar` file that contains all the required dependencies. This type of file is more simple to use than a `.war` file, which should be deployed on an external application server.

**Current topic:**

Getting started with Spring Boot    ⋯

**Topic depends on:**

✕  Introduction to Spring Boot    ⋯

✕  Gradle basics    ⋯

Topic is required for:

Basic project structure    ⋯

After you've chosen all the necessary settings and dependencies, just click **Generate Project** to get the project's archive. This archive contains the basic structure of a typical Spring Boot application without application logic; it is just a template, you need to write some logic there.

> Note that another way to generate a Spring Boot application is to use an [IDE like IntelliJ IDEA](#).

## §2. Running the application

Although the generated application is useless at the moment, it is a complete one. Let's run it! First, unpack the application and then visit the project's directory. In our examples, we will use Gradle as it is a common build automation system.

First, we should build the application:

```
1   ./gradlew build
```

It creates a `.jar` file using Gradle Wrapper and puts it to the `build/libs/` directory. Gradle automatically downloads all the necessary dependencies while building.

When finished, it should print:

```
1   BUILD SUCCESSFUL in 1s
```

Now, it is time to run the program represented as a `.jar` file.

```
1   java -jar build/libs/*.jar
```

After running, this application just prints the Spring logo and several log lines that we skipped.

```
1    .   ____          _            __ _ _
2   /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
3  ( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
4   \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
5    '  |____| .__|_| |_|_| |_\__, | / / / /
6   =========|_|==============|___/=/_/_/_/
7   :: Spring Boot ::        (v2.1.5.RELEASE)
```

You may also build and start an application using a single line:

```
1   ./gradlew build && java -jar build/libs/*.jar
```

Congratulations! You've just created and started your first application using the Spring Boot framework even without a single line of code, except for a pair of commands.

🗎 Report a typo

**151** users liked this theory. **4** didn't like it. **What about you?**

😍  🙂  😐  🙁  😡

**Start practicing**

Comments (17)          Hints (0)          Useful links (0)                              **Show discussion**