Appendix 03 Import Spring Tool Suite Project

Description: Not all Spring Tool Suites project are started from scratch within the IDE. There are many ready to run projects and shell projects that can be downloaded and imported into the IDE.

Project: This appendix shows a step by step guide to import a Spring Boot project from disk. The steps are generic so any Java based or Spring Framework projects can be imported.

Technology: This project uses the following technology:

- Integrated Development Environment (IDE):
  - Spring Tool Suite 4 (Version: 4.11.0.RELEASE)
- Java Development Kit (JDK):
  - Oracle's JDK 8 (1.8)

Table of Contents

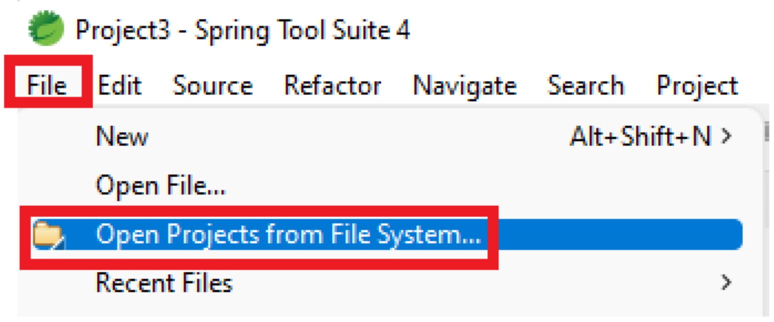
Import a Spring Tool Suite Project from Disk

Again any Java based or Spring Framework projects can be imported using the following steps. In these instructions we use The Spring Boot demo that was created using the document titled "**Appendix 02 Spring Initializr**". The project name is "demo-spring-boot-hello".

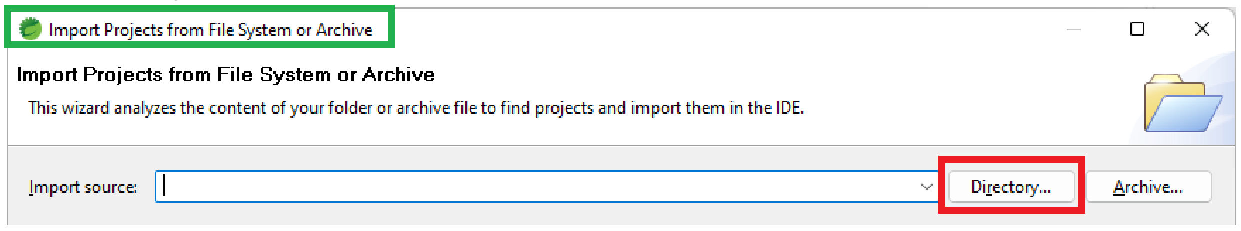
Import the project: "demo-spring-boot-hello"

Open the Spring Tool Suite 4 IDE.

From File Toolbar ➔ select "File" ➔ select "Open Projects from File System"



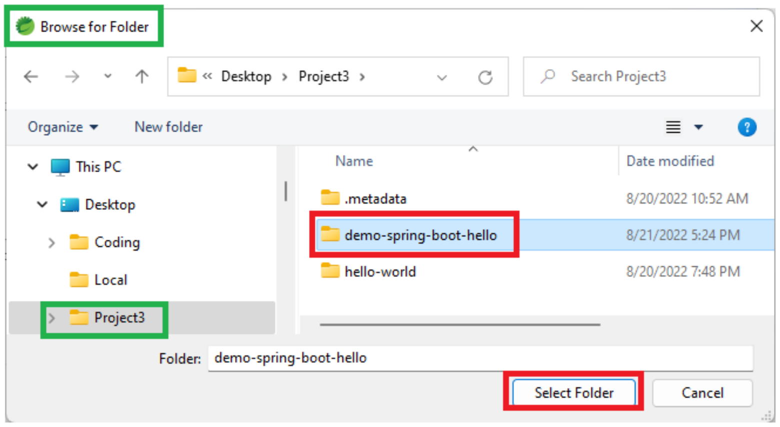
Click "Directory..."



Import Projects from File System or Archive

This wizard analyzes the content of your folder or archive file to find projects and import them in the IDE.

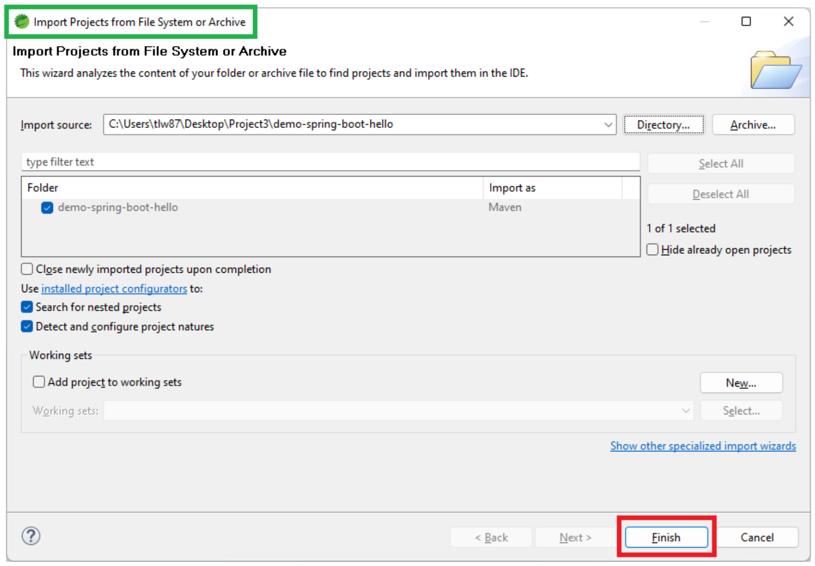
Navigate to top-level IDE workspace folder.
Select the project folder: "demo-spring-boot-hello"
Click "Select Folder"



Name	Date modified
.metadata	8/20/2022 10:52 AM
demo-spring-boot-hello	8/21/2022 5:24 PM
hello-world	8/20/2022 7:48 PM

Folder: demo-spring-boot-hello

Click “Finish”


Import Projects from File System or Archive

Import Projects from File System or Archive

This wizard analyzes the content of your folder or archive file to find projects and import them in the IDE.

Import source: C:\Users\tlw87\Desktop\Project3\demo-spring-boot-hello

Folder	Import as
☑ demo-spring-boot-hello	Maven

1 of 1 selected

☐ Hide already open projects

☐ Close newly imported projects upon completion

Use installed project configurators to:

☑ Search for nested projects

☑ Detect and configure project natures

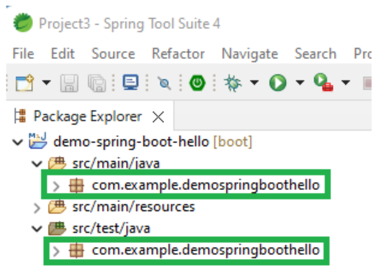
Working sets

☐ Add project to working sets

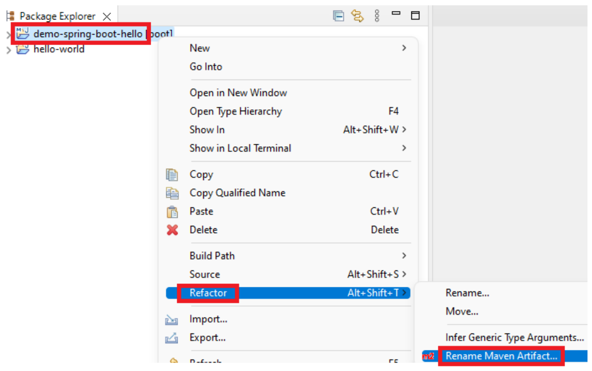
Spring Boot Project Structure

Any Java based or Spring Framework projects will have similar structure to a Spring Boot project. The main addition for a Spring Boot project created from the Spring Initializr website is the Spring Boot project contains a runnable jar file for a self-contained Tomcat server.

Expand the Project


Project3 - Spring Tool Suite 4
File Edit Source Refactor Navigate Search Pro
Package Explorer
demo-spring-boot-hello [boot]
src/main/java
com.example.demospringboothello
src/main/resources
src/test/java
com.example.demospringboothello

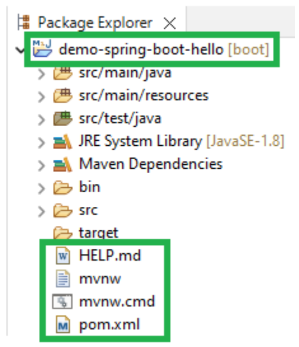
If your project structure does not look like the above and instead has folders for com, example, and others, try using the Maven artifact refactoring option. Right click on the project ➔ Refactor ➔ "Rename Maven Artifact…"



Examine the Spring Boot Project

The IDE project structure is similar to the folder structure we detailed in the document titled "**Appendix 02 Spring Initializr**". This is a typical Maven project structure.

Spring Boot Project Level Files



HELP.md - MD files are saved in plain text format that uses Markdown language. The HELP.md file is just that a help file or Read Me First file.

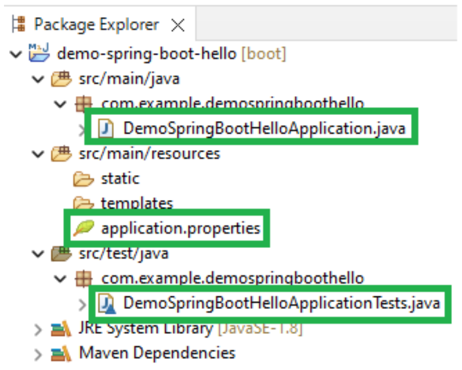
mvnw and **mvnw.cmd** - These are scripts designed to be used on the command line for your environment and will invoked the runnable jar file in the .mvn/wrapper directory. This will start the independent Tomcat server.

> **mvnw** - it's an executable Unix shell script used in place of a fully installed Maven. This calls the executable jar in the **.mvn\wrapper** folder.
>
> **mvnw.cmd** - it's for Windows environment. This calls the executable jar in the **.mvn\wrapper** folder.
>
> **.mvn** – folder not shown in the IDE containing the Maven wrapper files. Inside another subfolder "wrapper" is an executable jar file which will contain an Apache Tomcat server which Spring Boot application will run on. The Tomcat server inside the IDE will not be used in a Spring Boot application.

pom.xml - The Maven XML file that contains information about the project and configuration details used by Maven to build the project. It contains default values for most projects.

The Spring Initializr website generates a project with additional components added for Spring Boot and for convenience.

Spring Boot Project Package Level Files


```
Package Explorer
demo-spring-boot-hello [boot]
  src/main/java
    com.example.demospringboothello
      DemoSpringBootHelloApplication.java
  src/main/resources
    static
    templates
    application.properties
  src/test/java
    com.example.demospringboothello
      DemoSpringBootHelloApplicationTests.java
  JRE System Library [JavaSE-1.8]
  Maven Dependencies
```

In the src folder structure is the package we defined when creating the zip file: "com.example".

> **DemoSpringBootHelloApplication.java** – was generated as the class to run the application containing the public static void main(String[] args){} method.

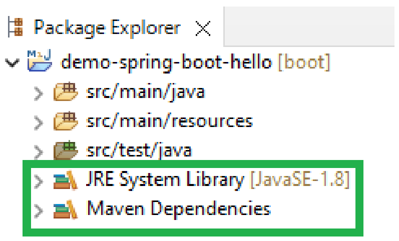
In the resource folder there is a property file shell, it is created as a placeholder, an empty file.

application.properties - This file contains the different configuration which is required to run the application in a different environment, and each environment will have a different property defined by it. It can contain database related properties such as connection string, username, password, and other properties.

A default test package was also created with “com.example” for unit testing.

DemoSpringBootHelloApplicationTests.java – was generated with class annotation @SpringBootTest to identify as a test class.

Spring Boot Project Library and Dependency Folders

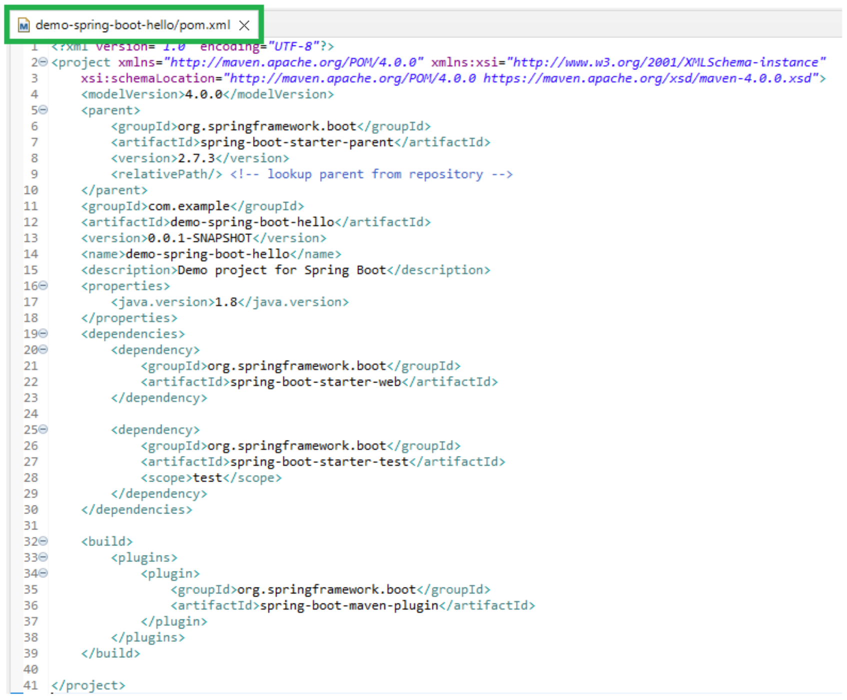


```
Package Explorer
demo-spring-boot-hello [boot]
    src/main/java
    src/main/resources
    src/test/java
    JRE System Library [JavaSE-1.8]
    Maven Dependencies
```

JRE System Library – contains all the Java runtime library required to run the Spring Boot project.
Maven Dependencies – contains all the imports for the dependencies define in the pom.xml.

Explorer the Spring Boot Project pom.xml File

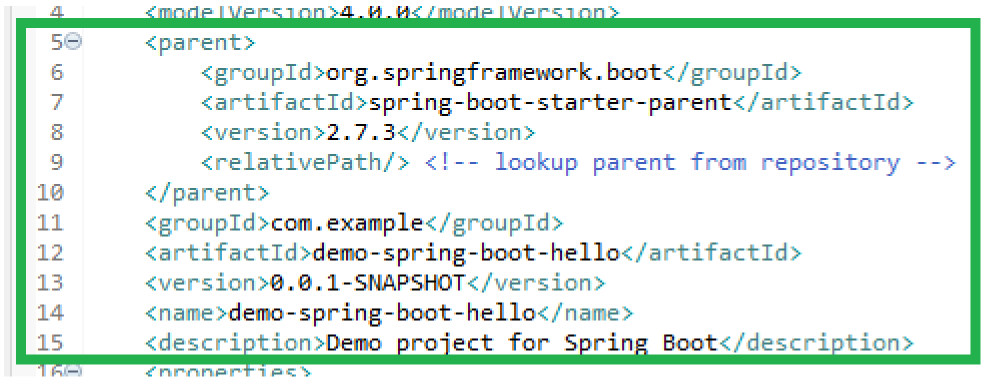
This is the pom.xml file generated for the “demo-spring-boot-hello” project. It is a basic Spring Boot project containing only the “Spring Web” dependency when generated using “Spring Initializr”.



demo-spring-boot-hello/pom.xml

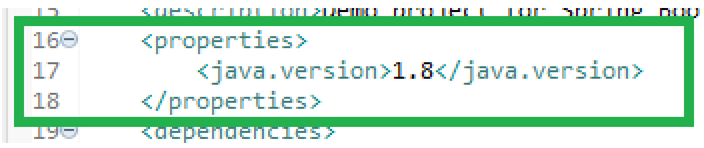
```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.7.3</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.example</groupId>
    <artifactId>demo-spring-boot-hello</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>demo-spring-boot-hello</name>
    <description>Demo project for Spring Boot</description>
    <properties>
        <java.version>1.8</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>

</project>
```

This area contains basic and metadata information entered when using “Spring Initializr”.



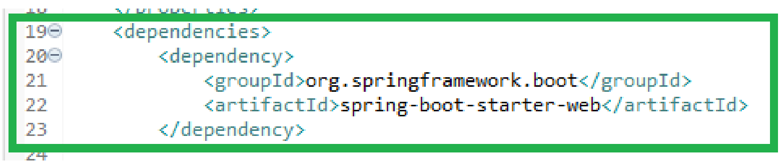
```xml
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.7.3</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.example</groupId>
    <artifactId>demo-spring-boot-hello</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>demo-spring-boot-hello</name>
    <description>Demo project for Spring Boot</description>
```

There is a properties section which shows which version of java is used.



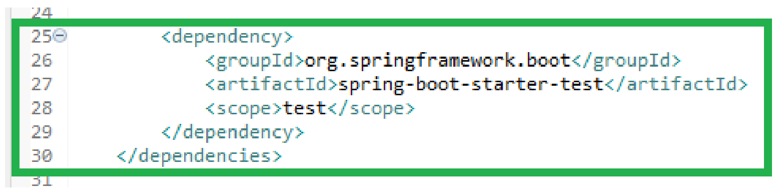
```xml
    <properties>
        <java.version>1.8</java.version>
    </properties>
```

This is the dependency add for “Spring Web”



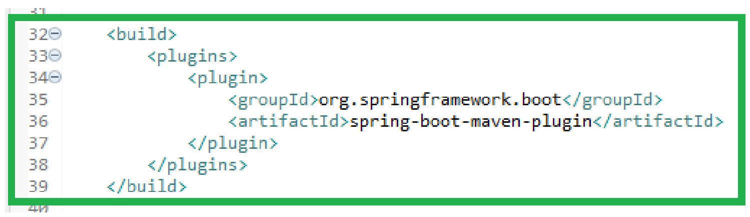
```xml
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
```

The other dependency is part of the simple Spring Boot framework.



```xml
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>
```

Finally information for building the Spring Boot application.



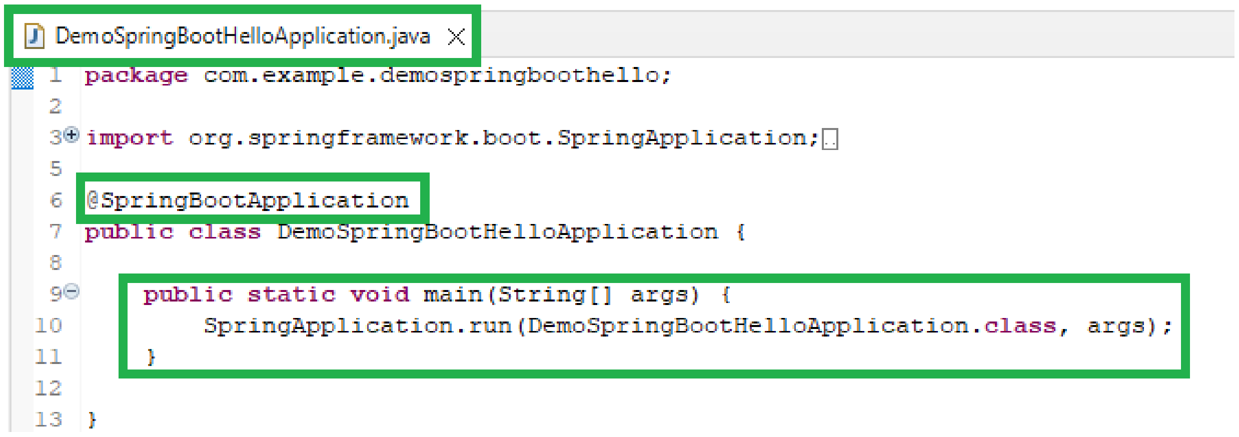
```xml
    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>
```

Depending on what dependencies are required in other projects using the “Spring Initializr”, the pom.xml will look different. Dependencies could be added or removed from what is shown in this section.

Open the Spring Boot Demo File

The Spring Initializr website generates a very basic shell “DemoSpringBootHelloApplication” to run the application. Other coaching documents like “Spring Boot 01 Hello World” will include details on steps to create projects related to the document.

Open the “DemoSpringBootHelloApplication”. The annotation `@SpringBootApplication` identifies this is a Spring Boot application. The statement `public static void main(String[] args) {}` is the runtime entry point for the application.



```java
package com.example.demospringboothello;

import org.springframework.boot.SpringApplication;

@SpringBootApplication
public class DemoSpringBootHelloApplication {

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

}
```

Once you have sufficient knowledge with Spring Framework, Spring Boot, RESTful APIs and other technologies, you can create your own projects based on the basic project structure generated by the Spring Initializr website.

Run the Spring Boot Project

The basic project generated by the Spring Initializr website is designed to be built and run from the command line. These projects can also be run inside the Spring Tool Suite IDE.

Instructions on running the Spring Initializr generated project is found in document titled: “Appendix 04 Run Spring Initializr Project”.