What is Spring Boot?

Spring Boot allows you to run Spring Framework web applications on an Application Server like Apache Tomcat or Jetty in a single jar file.

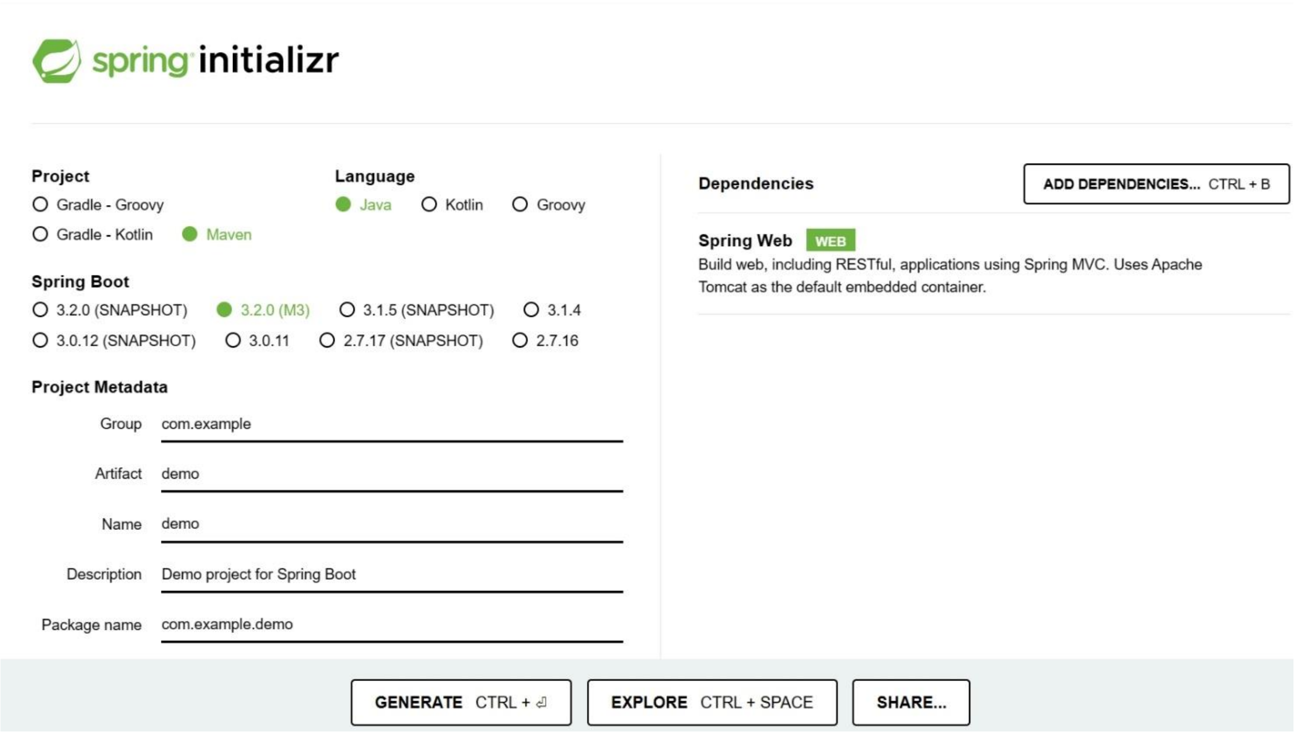
jar file contains your spring boot code for the web application as well as the application server like tomcat or jetty embedded within the executable jar file of the spring boot application.

This tutorial will guide you through the process of making a spring boot application.

We will be using IntelliJ IDEA (Community Edition) as IDE and we will be using website

https://start.spring.io for making a skeleton or an initial spring boot project

go to start.spring.io and make a demo spring boot project as shown below



Project
○ Gradle - Groovy
○ Gradle - Kotlin ● Maven

Language
● Java ○ Kotlin ○ Groovy

Spring Boot
○ 3.2.0 (SNAPSHOT) ● 3.2.0 (M3) ○ 3.1.5 (SNAPSHOT) ○ 3.1.4
○ 3.0.12 (SNAPSHOT) ○ 3.0.11 ○ 2.7.17 (SNAPSHOT) ○ 2.7.16

Project Metadata

Group: com.example

Artifact: demo

Name: demo

Description: Demo project for Spring Boot

Package name: com.example.demo

Dependencies

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

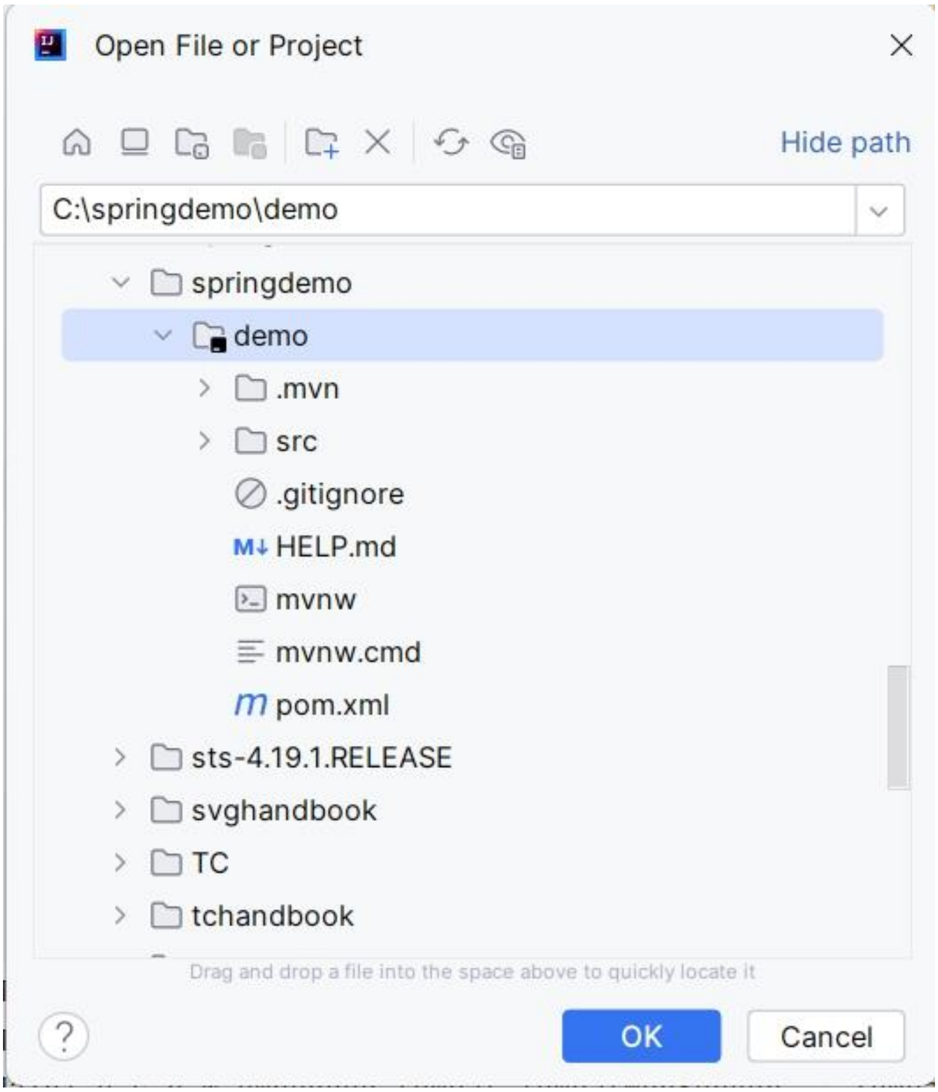
Now Click on Generate and download the zip file of the demo spring boot project

unzip the zip file to a folder c:\springdemo and open it in IntelliJ IDEA as shown below

You can download IntelliJ IDEA Community Edition from following link

Download IntelliJ IDEA – The Leading Java and Kotlin IDE (jetbrains.com)

Now Start IntelliJ IDEA and open the project in the IDE

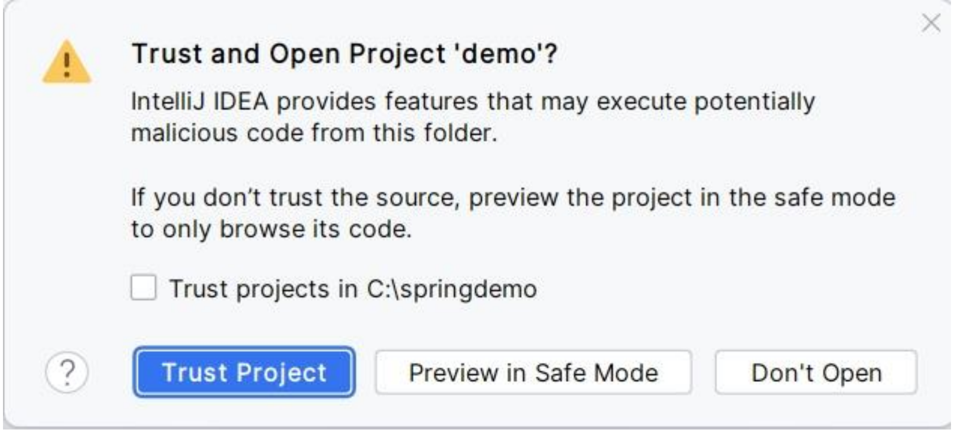


C:\springdemo\demo

Drag and drop a file into the space above to quickly locate it

Click on OK

Following window will appear



Trust and Open Project 'demo'?

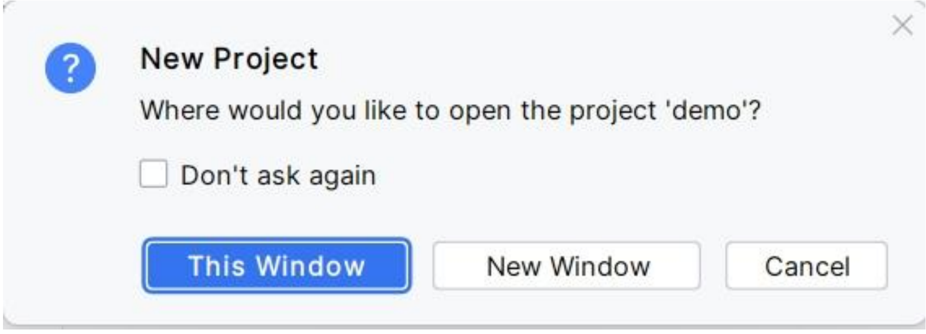
IntelliJ IDEA provides features that may execute potentially malicious code from this folder.

If you don’t trust the source, preview the project in the safe mode to only browse its code.

□ Trust projects in C:\springdemo

Click on Trust Project

Following window will appear



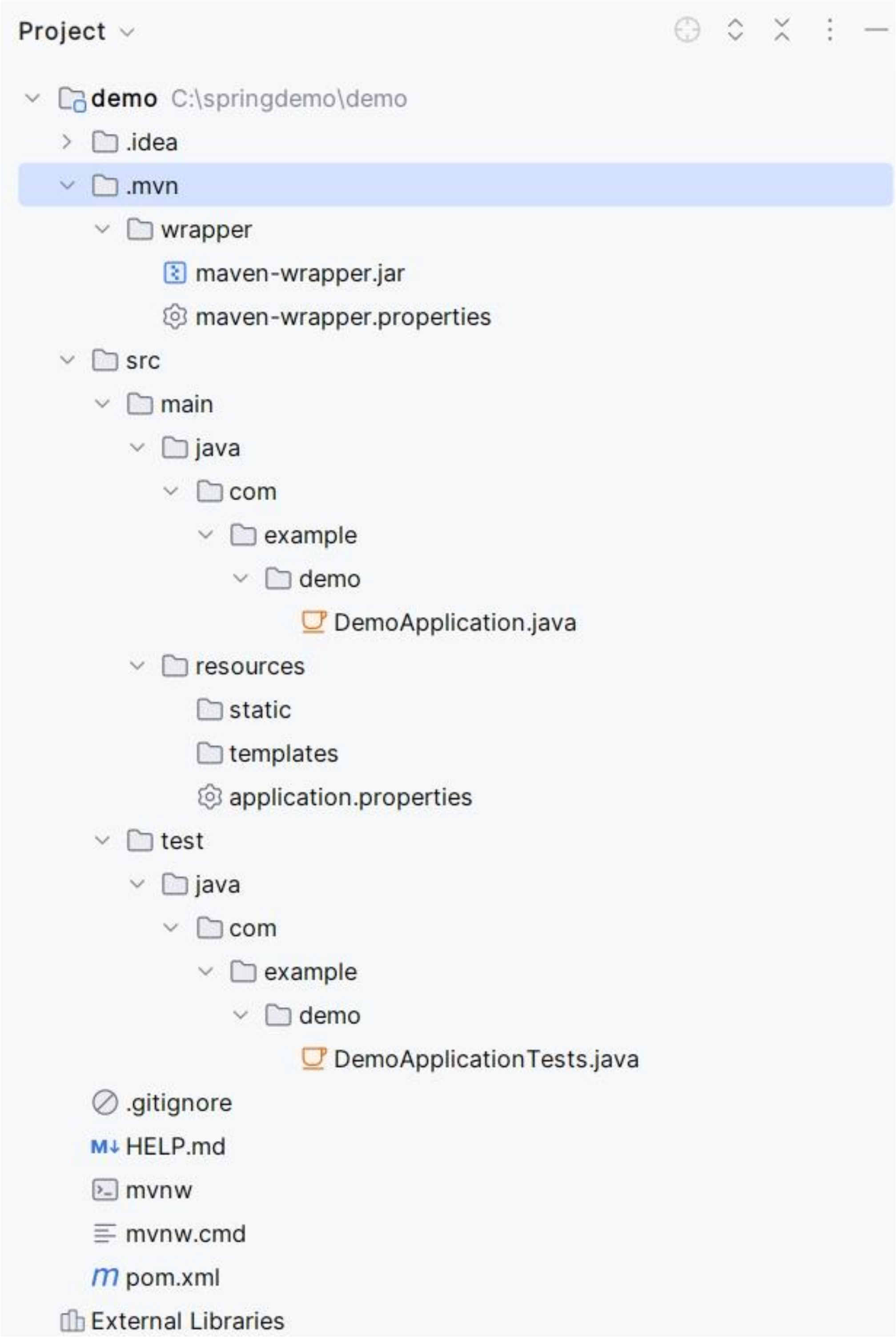
New Project

Where would you like to open the project 'demo'?

□ Don't ask again

Click on This Window and project will be opened in the IDE.

Following is the directory structure of the project



```
Project
demo C:\springdemo\demo
  .idea
  .mvn
    wrapper
      maven-wrapper.jar
      maven-wrapper.properties
  src
    main
      java
        com
          example
            demo
              DemoApplication.java
      resources
        static
        templates
        application.properties
    test
      java
        com
          example
            demo
              DemoApplicationTests.java
  .gitignore
  HELP.md
  mvnw
  mvnw.cmd
  pom.xml
External Libraries
```

This project is a Maven project.

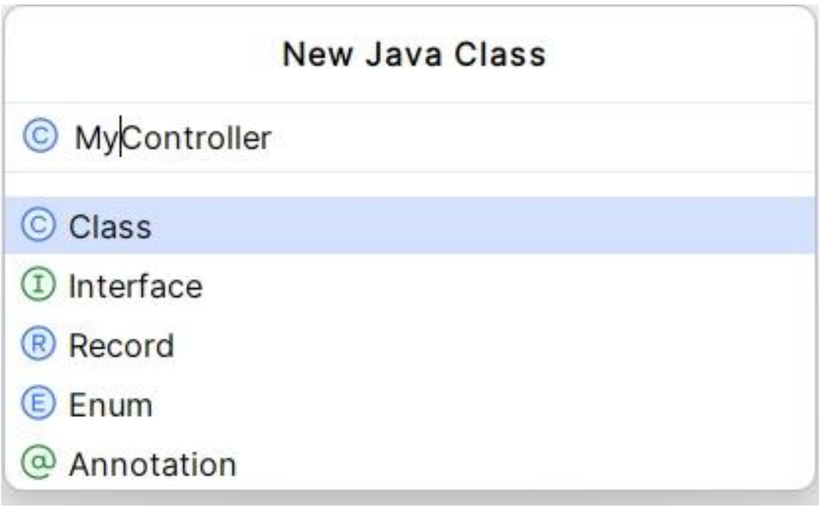
To know more about a Maven project take help from handbook on cppcourses.com

http://www.cppcourses.com/mavenhandbook.pdf

Now we will make a rest controller in the project

Rest Controller is a controller as per MVC Design Pattern and maps Mappings in the web application

Now right click on demo folder and make a new Class MyController as shown below



MyController

Write the following code in MyController Class

```java
package com.example.demo;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
@RestController
public class MyController
{
    @GetMapping("/")
    public String sayHello()
    {
        return "Hello World!";
    }
    @GetMapping("/hello")
    public String sayHello1()
    {
        return "Hello abcdef";
    }
}
```

@RestController and @GetMapping are annotations.

Annotation in java is a metadata about a class, object, method.

metadata means information about a class, object, method.

@RestController annotation means MyController class is a RestController Class

and @GetMapping annotation means sayHello method will point to path / and sayHello1 will point to path /hello.

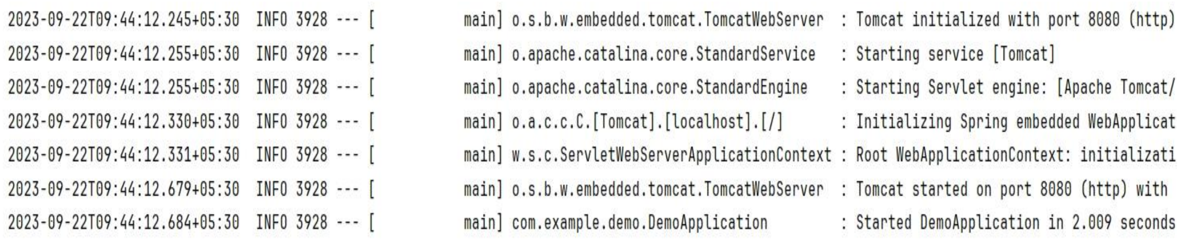
Now save the file

and run the project as mentioned below

right click on DemoApplication and click on Run 'DemoApplication.main()'

This will run embedded Apache Tomcat Server and the spring boot application.

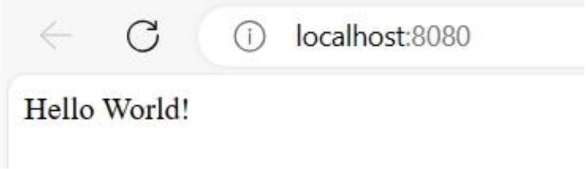
you will see following output in the terminal.



```
2023-09-22T09:44:12.245+05:30  INFO 3928 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port 8080 (http)
2023-09-22T09:44:12.255+05:30  INFO 3928 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2023-09-22T09:44:12.255+05:30  INFO 3928 --- [           main] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/
2023-09-22T09:44:12.330+05:30  INFO 3928 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicat
2023-09-22T09:44:12.331+05:30  INFO 3928 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initializati
2023-09-22T09:44:12.679+05:30  INFO 3928 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http) with
2023-09-22T09:44:12.684+05:30  INFO 3928 --- [           main] com.example.demo.DemoApplication         : Started DemoApplication in 2.009 seconds
```

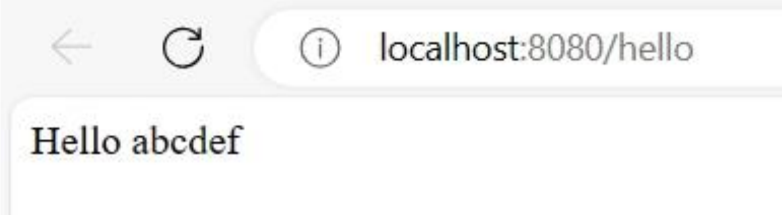
Open your browser and point to http://localhost:8080/

You will see following output



Hello World!

Now point your browser to http://localhost:8080/hello



Hello abcdef

Now the topic is application.properties file.

application.properties file contains properties of your application and it is present in resources folder.

Now edit application.properties file and add the following lines to it.

```
my.name=Raman
```

Now to get a value from application.properties file we use annotation @Value

Add the following import to MyController Class

```
import org.springframework.beans.factory.annotation.Value;
```

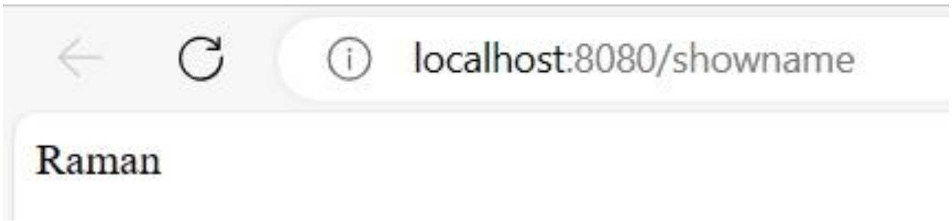
and also add following code to MyController Class

```
@Value("${my.name}")
private String myname;

//Above code will save property my.name into String object myname

@GetMapping("/showname")
public String showname()
{
    return myname;
}
```

Now again run your project

and point your browser to http://localhost:8080/showname

You will get the following output



Raman