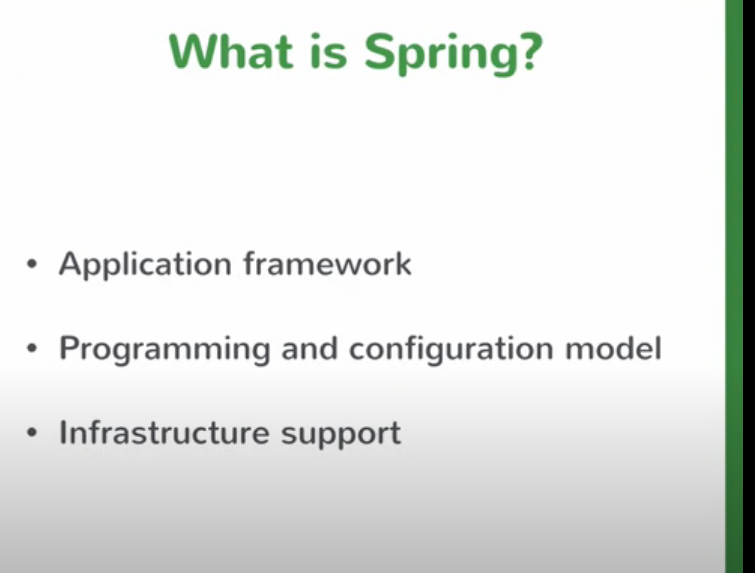
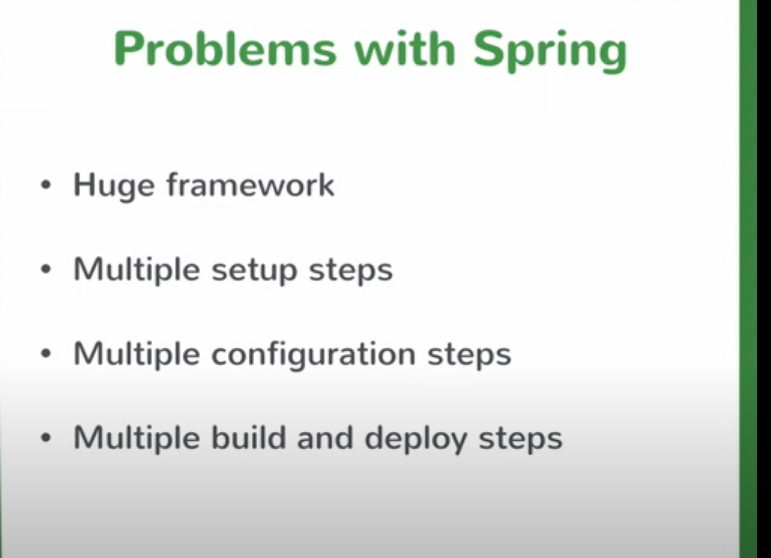
Spring Boot





Spring was developed in order to solve common problems which developers face while creating enterprise-level application like transaction management. Spring helps you to build enterprise-level java application. It lets you connect to different kinds of databases.



It has become a huge framework. It needs a lot of configuration to start the application. In order to connect to different types of databases different configurations need to be done. There are multiple build and deployment steps.

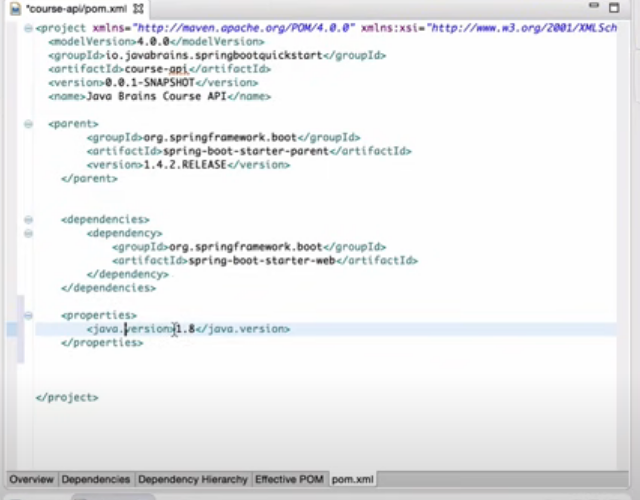
Spring-based applications can be built in many different ways. These ways depend on the type of application to be created. There is no single way to create a spring-based application and there is no sure shot way to create spring based application that will work in majority of cases. Here, spring boot comes into picture.

Spring boot helps in creating spring based application , and this way of creating application works for majority of cases.



It works for 80% of the use cases; if any changes are required, they can be made later. It helps in creating the production-ready application i.e we don’t have to make changes to it in order to deploy it to production.

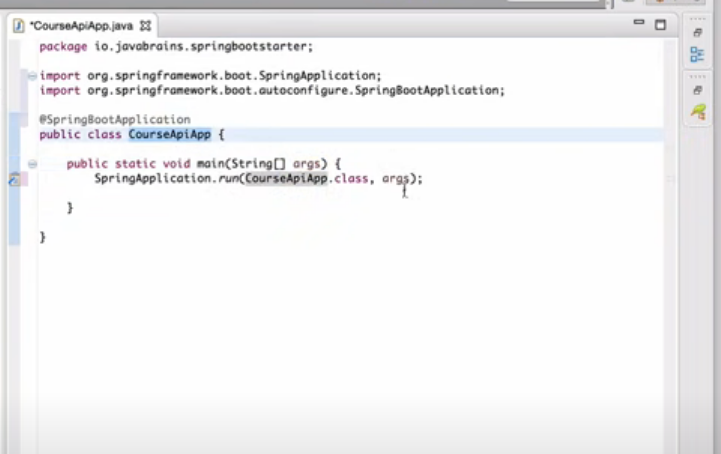
Creating Maven project



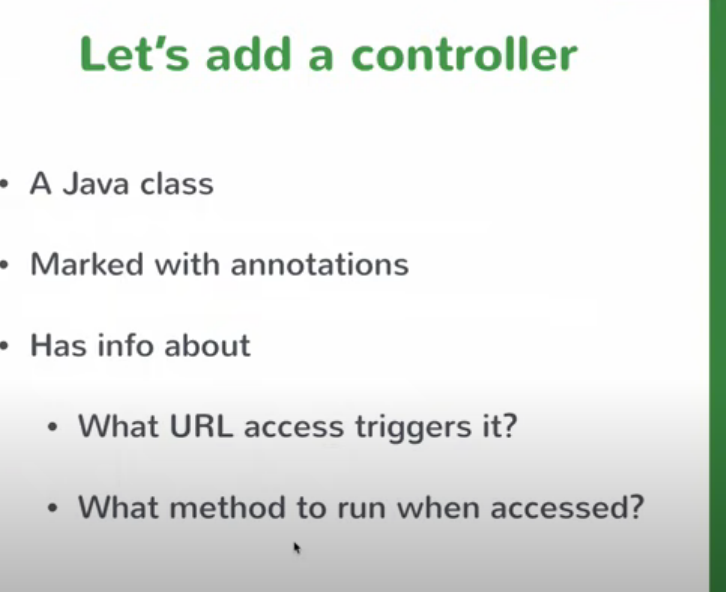
The parent tag in the above screenshot means that our project is the child of the parent project i.e spring. Spring-boot-starter-web is a dependency that is needed to create a web-based application. It is a combination of different dependencies which are required to build web-based applications.

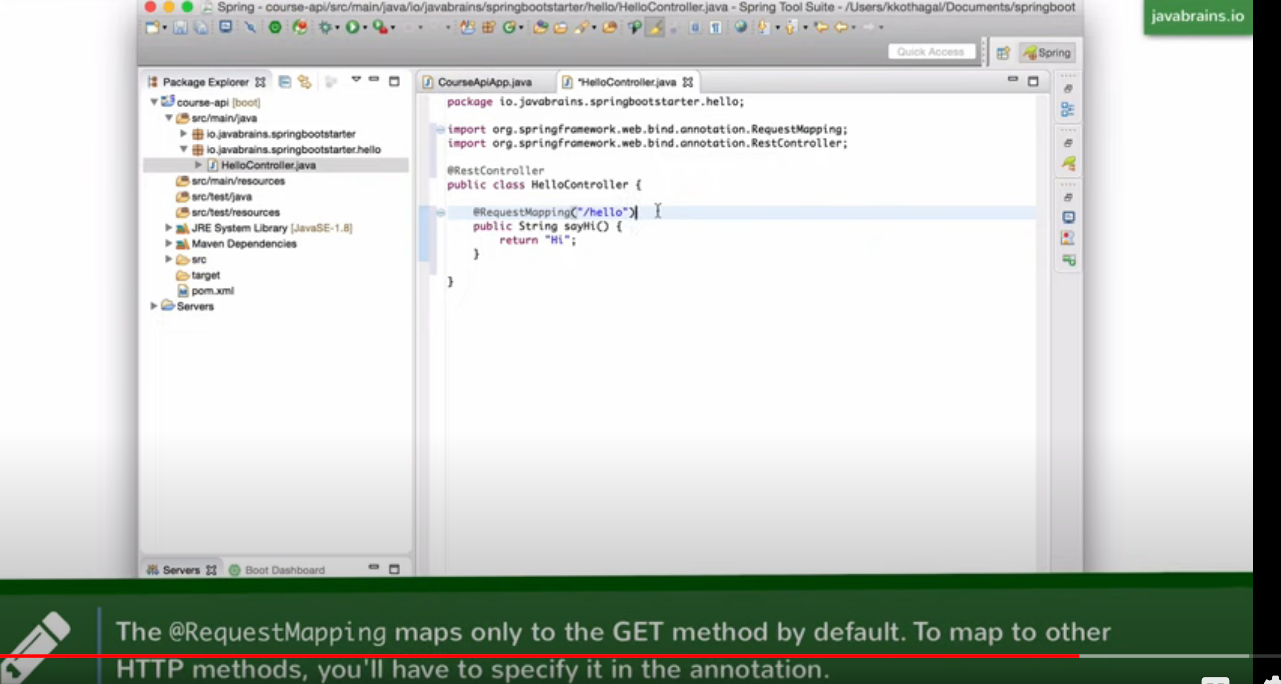
Spring boot starter parent dependency is of a specific version and if we change it’s version then it’s relevant dependencies version also gets changed.

@SpringBootApplication tells spring boot that it is the starting point of spring boot application.

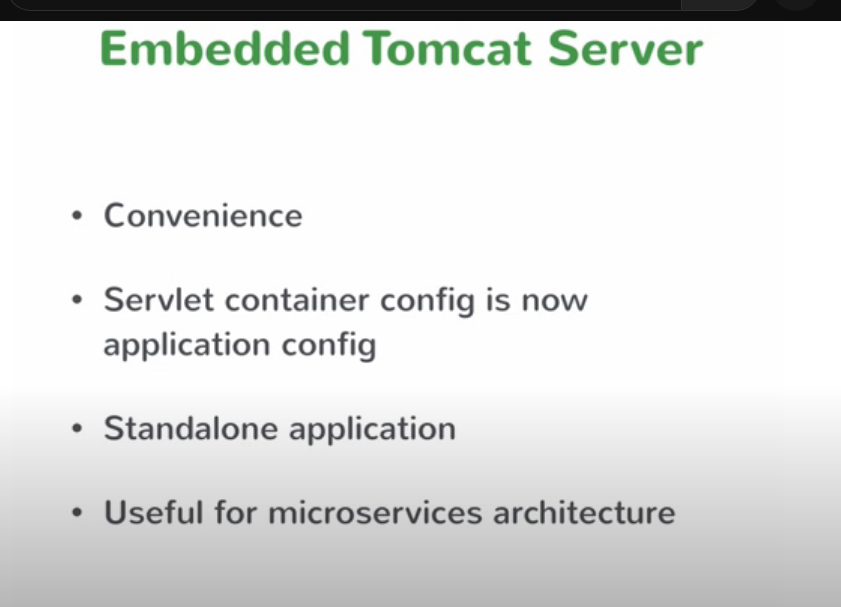


In order to start spring boot application we call the static run method of SpringApplication and pass the class name that executes this method as argument , we may also pass String agrs to the SpringApplication class.





In spring boot we don’t have to download any server(like tomcat) to deploy our local application. It is by default added to spring boot project i.e it has an embedded tomcat server.



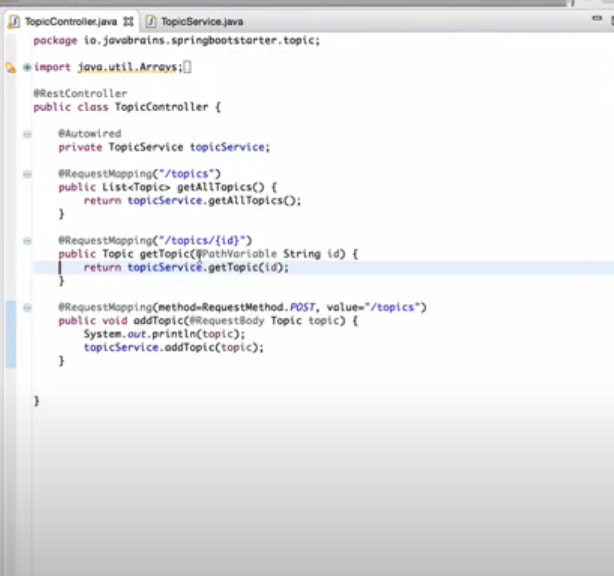
Embedded tomcat server in Spring boot helps in creating standalone applications which can be deployed easily, It does not require any servlet container/servlet mapping configuration which we usually do in web.xml file.

It Is also useful in microservice architecture.

@Service

It is used to create business service classes.

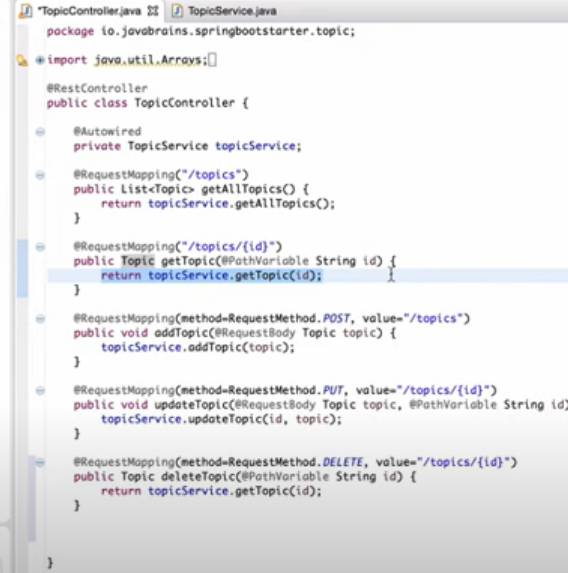
@RequestBody is used to take object as an input (specially in case of POST request).



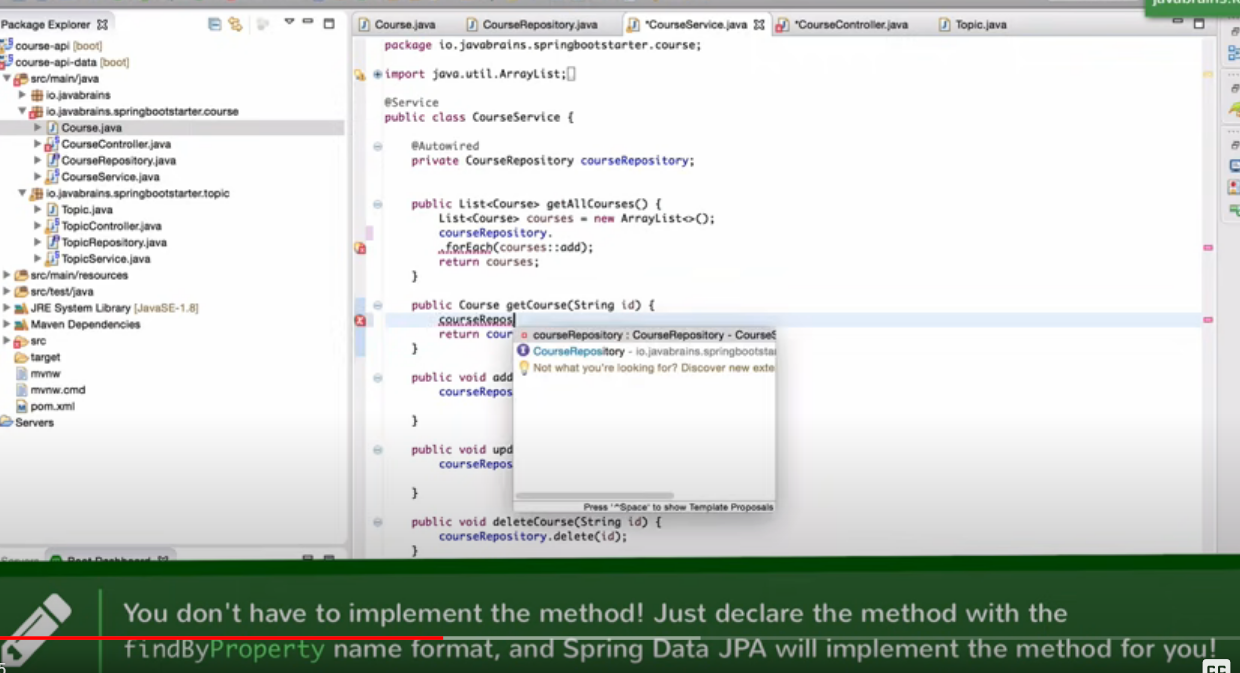
POST VS PUT

Important difference between the methods is that **PUT is an idempotent method, while POST isn't**. For instance, calling the PUT method multiple times will either create or update the same resource. In contrast, multiple POST requests will lead to the creation of the same resource multiple times.

<https://www.baeldung.com/rest-http-put-vs-post#:~:text=Another%20important%20difference%20between%20the,the%20same%20resource%20multiple%20times>.



While using Spring data jpa , if you want to define new methods then we don’t have to create it’s implementation, we just have to create method name in a specific format and spring automatically creates it implementation.



In order to deploy a spring boot application, we first need to create a jar file.

So, first we’ll do clean install that will create a jar file in the target folder and we’ll run/execute that jar file by using the below command.

Java – jar target/JAR\_FILE\_NAME

We may also change the packaging from jar to war in the pom.xml file.

**Spring Boot Actuator**

Actuator brings production-ready features to our application.

Monitoring our app, gathering metrics, understanding traffic, or the state of our database become trivial with this dependency.

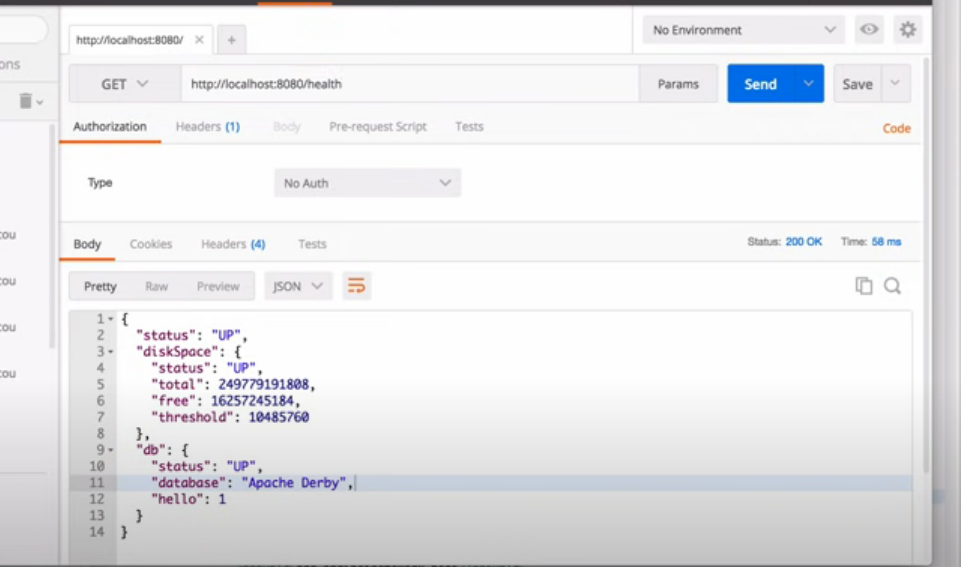
The main benefit of this library is that we can get production-grade tools without having to actually implement these features ourselves.

Actuator is mainly used to expose operational information about the running application — health, metrics, info, dump, env, etc. It uses HTTP endpoints or JMX beans to enable us to interact with it.

Once this dependency is on the classpath, several endpoints are available for us out of the box. As with most Spring modules, we can easily configure or extend it in many ways.

It creates a new endpoint with /health route. We may also change it’s port number in application.properties by using properties

management.port = 9091(or any other port number)



Use this dependency

<**dependency**>

<**groupId**>org.springframework.boot</**groupId**>

<**artifactId**>spring-boot-starter-actuator</**artifactId**>

</**dependency**>

We can also use HAL explorer for better visualization of the links/details provided by Actuator.