# Spring Boot Basics

## Spring Boot

### DEFINITION/WHAT

Built on top of the conventional spring framework.

It is developed by Pivotal Team

### NEED/WHY

Need 1 : It is easy to create a stand-alone and production ready spring applications .

Need 2 : Supports embedded servers

Need 3 : Built-in auto-configuration capabilities, it automatically configures both the underlying Spring Framework and third-party packages based on your settings

### IMPLEMENTATION/HOW IT INTERNALLY WORKS

Spring Boot automatically configures your application based on the dependencies you have added to the project by using annotation

The entry point of the spring boot application is the class that contains @SpringBootApplication annotation and the main method.

Spring boot main() method calls static run() which is present in Spring Application

### ALTERNATIVE/DIFFERENCES

Difference in spring and spring boot

| Spring Framework | Spring Boot |
|---|---|
| Is an open-source lightweight framework widely used to develop enterprise applications | is built on top of the conventional spring framework, widely used to develop REST APIs |
| It helps to create a loosely coupled application | It helps to create a stand-alone application |
| The most important feature of the Spring Framework is dependency injection | The most important feature of the Spring Boot is Autoconfiguration |
| To run the Spring application, we need to set the server explicitly. | Spring Boot provides embedded servers such as Tomcat and Jetty etc |
| It doesn't provide support for the in-memory database | It provides support for the in-memory database such as H2 |
| To run the Spring application, a deployment descriptor is required | There is no requirement for a deployment descriptor |

## ADDITIONAL INFORMATION

There are Four ways using which we can create spring boot project

1) Create a maven project and add the starter dependencies
2) Use spring Initializer
3) Use the IDE (Spring Tool Suit or IntelliJ)
4) Using Spring boot CLI

# Spring Initializer

## Definition/What

A web application that helps you to create an initial spring boot project structure

Provides a maven or gradle file to build your code

## Need/Why

Need 1 : A convenient way to create a Spring Boot project

Need 2 : Creates a skeleton project for us and saves setup time so that we can concentrate on adding business logic

## Implementation/How it internally works

Go to the Spring Initializr site -https://start.spring.io/

Choose a dependency management tool (either Maven or Gradle), a language (Java, Kotlin or Groovy), a packaging scheme (Jar or War), version and dependencies, and download the project.

Even when we use our IDE's (such as STS or Eclipse with STS plugin) new project wizard to create a Spring Boot project, it uses Spring Initializr under the hood.

# Spring Boot Annotations

## @SpringBootApplication

### Definition/What

The java class annotated with @SpringBootApplication is the main class of a Spring Boot application and application starts from here.

### Additional Information

It is a combination of three annotations @EnableAutoConfiguration, @ComponentScan, and @Configuration.

## @EnableAutoConfiguration

### Definition/What

enable auto-configuration mechanism.

## @ComponentScan

### Definition/What

searching packages for Components

The process of discovering Spring Components within classpath of the application

### Implementation/How it internally works

@ComonentScan without attribute tells that Spring to discover components in current package and within it's sub packages.

@ComonentScan with attribute tells that Spring to which packages to scan specifically by using basePackages attribute

# @Configuration

## Definition/What

The class annotated with @Configuration used by Spring Containers as a source of bean definitions.

## Implementation/How it internally works

@Configuration annotation indicates that a class declares one or more @Bean methods and may be processed by the Spring container to generate bean definitions and service requests for those beans at runtime

## Real Time Example

```
@Configuration
public class HelloWorldConfig {

@Bean
 public HelloWorld helloWorld(){
            return new HelloWorld();
}}
```