OOP, classes, interfaces, collection framework, inheritance, exception handling

SpringBoot

Why?

To build java applications.

Must’ s

1. JDK 17 or above -> Springboot 3
2. IntelliJ IDE

The Problem with spring:

Qs:

1. How do I set up the configuration? (xml or java)
2. Which JAR dependencies do I need?
3. How do I install the server?(Tomcat or JBoss or etc)
   & this is just getting started?

The Solution on this is SpringBoot

- Easier for development
- Provides the embedded server
- It resolves the dependency conflicts
- It provides auto-configuration

- Spring and SpringBoot
  SpringBoot uses Spring BTS
  SB makes it(development) easier for us

- Spring Initializer(Provided by SpringBoot)
  Start.spring.io

- Quickly creates a starter spring project
- Select dependencies
- Select build tool(maven / gradle)
- Import the project into IDE

SpringBoot embedded server:

Provide the embedded server

- Tomcat, JBoss, Undertow

No need to install server manually

FAQs

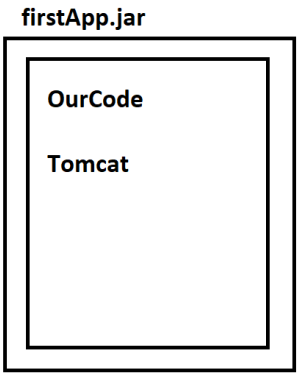
1. Does SB run code faster than regular spring code?
   ➔No, SB uses same code of spring framework.

2. Does SpringBoot replaces Spring MVC, Spring REST, ……?
   ➔No, It uses these technologies.

Maven:

When building our project, we may require additional JAR.

Ex. Hibernate, JSON, Spring, etc



JAR files includes our application code & the embedded server
JAR is a self-contained unit.

1st Approach

Download the JAR files manually from each project website

Manually adding JAR files to our classpath

2nd Approach-> Maven is the solution

- Tell maven the projects we are going to work on(dependencies)
- Maven will go out and download the JAR file for our project.
- Maven will make these JAR files available during the execution.
- Maven is like our personal helper or shopper(shopping list)

Development Process:

1. Configure the project at spring initializer (dependency: Spring Web)
2. Download the zip file
3. Unzip it
4. Import project into our IDE
5. Run the project

We have not added any real code to our project that's why we are getting this page.

6. Create a package -> rest
7. Create a class FunRestController
8. Add the @RestController Annotation on the class
9. Create a method which will return string("PrasadJain")
10. Annotate the method @GetMapping("/name")
11. Run the application

```java
package com.flynaut.exploration.rest;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class FunRestController {

    //exposing the endpoint - "/name" which will return name
    @GetMapping("/name")
    public String sayName(){
      return "PrasadJain";
    }

    //To return the today's date
    //To return yesterday's date
    //TASK

}
```

URL- Uniform Resource Locator

http://localhost:8080

http://www.google.com:8080/college

http: Application Layer Protocol

www.google.com: DNS qualified hostname/IP address(used to resolve the host problem)

8080: TCP port(used to identify the port)

/college: URI (Uniform Resource Identifier)or path or endpoint.