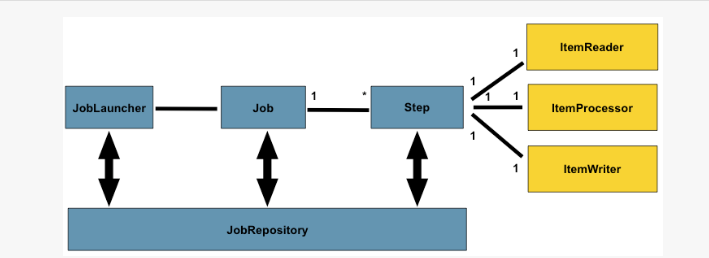
<https://spring.io/projects/spring-boot>

<https://spring.io/quickstart>

**Spring Batch:** Spring Batch. Spring Batch is a processing framework designed for robust execution of jobs.



Spring Batch provides classes and APIs to read/write resources, transaction management, job processing statistics, job restart and partitioning techniques to process high-volume of data.

* A job can consist of ‘n’ number of steps. Each step contains **Read-Process-Write** task or it can have single operation, which is called tasklet.
* Read-Process-Write is basically read from a source like Database, CSV etc. then process the data and write it to a source like Database, CSV, XML etc.
* Tasklet means doing a single task or operation like cleaning of connections, freeing up resources after processing is done.
* Read-Process-Write and tasklets can be chained together to run a job.

How Spring Batch Can Help Us?

A Spring Batch provides the following features that help us to solve multiple problems:

* It helps developers to structure the code in a clean way by providing the infrastructure that is used to implement, configure, and run batch jobs
* It uses the *chunk oriented processing* where items are processed one by one and the transaction is committed when the chunk size is met. In other words, it provides developers an easy way to manage the size of the transactions
* It provides the proper error handling. For e.g., developers can skip items if an exception is thrown and configure the retry logic that is used to determine whether the batch job should retry the failed operation. Developers can also configure the logic that is used to decide whether or not our transaction is rolled back
* It writes the comprehensive logs in the database. These logs contain the metadata of each job execution and step execution, and developers can use it for the troubleshooting purposes

**static** Job *jobObj*;

**static** JobLauncher *jobLauncherObj*;

* JobLauncher accepts Job and Job Parameters

JobExecution execution = *jobLauncherObj*.run(*jobObj*, **new** JobParameters());

* ItemProcessor accepts input,output

ItemProcessor<Report, Report>

* Writing business logic in a chunk-oriented step
* Processing items in a chunk-oriented step
* Transforming items
* Filtering items
* Validating items

**What is Spring Boot? 2.5.5 latest version**

Spring Boot is a tool which helps to build spring based standalone and production ready, enterprise Projects.

No need of servlet container to run application, as it comes with its own embedded tomcat.

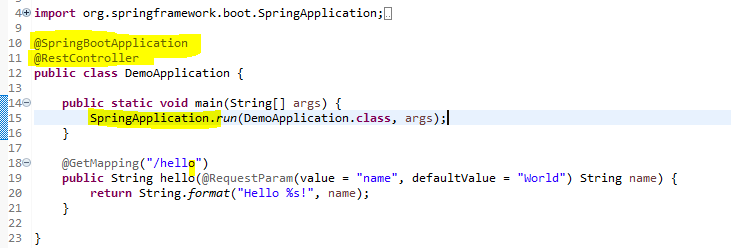
It has opinion how the configuration should be, which helps 80% of the time spring configuration set up things.

**Spring:** is wide framework used to build application much faster without worrying about configuration and infrastructure. We can focus on actual business implementation.

|  |  |
| --- | --- |
| **Spring** | **Spring Boot** |
| **Spring Framework** is a widely used Java EE framework for building applications. | **Spring Boot Framework** is widely used to develop **REST APIs**. |
| It aims to simplify Java EE development that makes developers more productive. | It aims to shorten the code length and provide the easiest way to develop **Web Applications**. |
| The primary feature of the Spring Framework is **dependency injection**. | The primary feature of Spring Boot is **Autoconfiguration**. It automatically configures the classes based on the requirement. |
| It helps to make things simpler by allowing us to develop **loosely coupled** applications. | It helps to create a **stand-alone** application with less configuration. |
| The developer writes a lot of code (**boilerplate code**) to do the minimal task. | It **reduces** boilerplate code. |
| To test the Spring project, we need to set up the sever explicitly. | Spring Boot offers **embedded server** such as **Jetty** and **Tomcat**, etc. |
| It does not provide support for an in-memory database. | It offers several plugins for working with an embedded and **in-memory** database such as **H2**. |
| Developers manually define dependencies for the Spring project in **pom.xml**. | Spring Boot comes with the concept of **starter** in pom.xml file that internally takes care of downloading the dependencies **JARs** based on Spring Boot Requirement. |

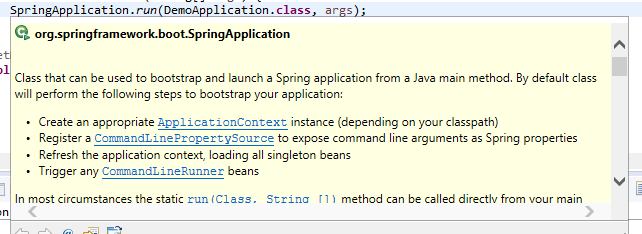
**Spring STS (Spring Tool Suite 4):** [**https://spring.io/tools**](https://spring.io/tools)

Same flavor as eclipse.



Launcher class should be annoted with **@SpringBootApplication**

**SpringApplication** : class that can be used to bootstrap and launch a Spring application from a Java main method. By default class will perform the following steps to bootstrap your application:



**Bill Of Material(BOM)** : is a concept which identifies all required configuration and dependencies require to run spring boot project, and as part of that adding parent in pom downloads all require dependencies for mentioned version along with compatible version of other dependencies.

**Spring Maven Project**

**Spring Initializer**

**Spring Boot CLI**

**Spring STS**

**Application.properties:** As we know Spring Boot is opinionated for configuration and has some autoconfig support, and for same it does have some keys, which we can define in application.properties.

<https://docs.spring.io/spring-boot/docs/current/reference/html/application-properties.html>

**Spring Data JPA:** is another spring project which supports data persistence and ORM for Spring Project.

