



Suresh Kumar

Follow

Feb 13 · 4 min read · Listen



Save



Sign in to Medium with Google



Saratha Natarajan

sarathanatarajan17@gmail.com



Saratha Poovalingam

saruusa@gmail.com

Spring Boot Starters, classes, interfaces Cheat Sheet

In my previous post on [Spring Boot Annotations](#), we have seen commonly used Spring, Spring Boot Annotations. In this post lets explore the commonly used Spring, Spring Boot maven starter dependencies, classes and interfaces(excluding annotations).

Open in app ↗

Sign up

Sign In



Classes interfaces

Spring Boot Starters, Classes, interfaces

As known, we can start using Spring, Spring Boot in Java application, by adding related Spring, Spring Boot starter dependencies in pom.xml(for Maven Projects).



Below are most commonly used Spring Boot dependencies.

a) **spring-boot-starter-web** — includes Spring Core, MVC, REST API related dependencies, including Apache Tomcat Web Server

- b) **spring-boot-starter-data-jpa** — includes Hibernate ORM and related Spring Boot functionality to interact with RDBMS databases
- c) **spring-boot-starter-validation** — provides validation related functionality
- d) **spring-boot-starter-test** — provides Test related functionality
- e) **spring-boot-starter-aop** — provides Aspect Oriented programming
- f) **spring-boot-starter-security** — provides Authentication, Authorization, Security Filters
- g) **spring-boot-starter-actuator** — provides actuator which is useful in remote health monitoring of running Spring Boot application
- h) **spring-boot-devtools** — provides remote debugging, automatic live reload of project, whenever source code changes are saved (this saves development time, by avoiding manual restart of running project)
- i) **spring-boot-starter-webflux** — Used to develop reactive, non-blocking REST API end points

There are many other such Spring boot starter dependencies, for example for interacting with Mongo DB, Kafka Messaging, etc...

A new Spring boot application can be created by

1. visiting Spring Initializr(<https://start.spring.io>)  3 |  and selecting required dependencies (i.e. starter packages), and download sample Project, to which developers can add required business logic.
2. STS (Spring Tool Suite) provides default support to create Spring Boot application by adding required dependencies, such as above. For this, other IDEs such as Eclipse, IntelliJ need appropriate plugin to be installed.
3. other way is, such starter dependencies can be added manually, to a Maven project.

Spring Boot Classes, interfaces: Once starter dependencies such as above are added, Spring Boot framework can be used in your Applications either by

- a) specifying appropriate Annotations (these we have explored in my previous blog post)

b) using classes or interfaces (commonly used Spring boot classes or interfaces are mentioned below)

c) using Configurations in application.properties (I will be briefing this in my next posts)

Lets explore Spring, Spring Boot's classes or interfaces

BeanFactory & ApplicationContext — These classes are generally used to specify Configuration classes and to explicitly request creation of a bean, these both classes are bit rarely used (directly), in application development.

ResponseEntity<> — a Generic class, which bundles the HTTP response sent back to Client. With ResponseEntity you can specify

a. Http status code

b. Http Headers

c. Response Body

CommandLineRunner — interface with run() method declaration — override run() with functionality which need to be executed, immediately after Spring Container is ready.

UriBuilder — to create flexible urls along with Path Variables, Query Parameters. These Urls are used, to further invoke REST APIs, from Spring Boot Application.

CrudRepository — interface with methods declared, that facilitates application to interact with any RDBMS database to perform CRUD operations

PagingandSortingRepository — interface which is derived from CrudRepository, and additionally provides method declarations for paging and sorting related functionality

JpaRepository — interface which is derived from CrudRepository, additionally provides method declarations for Paging and sorting related functionality, and batch related operations

Pageable, Page, Sort — classes or interfaces which lets to customize Sorting and Paging functionality, such as ascending, descending, page size, sort based on

multiple criteria, etc...

WebSecurityConfigurerAdapter — derived class of this class, is used to specify Security related Configurations

AuthenticationManagerBuilder — used to specify source of Authentication details, type of Authentication source, encryption algorithm.

SecurityContext, SecurityContextHolder — SecurityContext class holds details of currently authenticated(or logged in) User. And SecurityContextHolder is helper class to provide us SecurityContext.

HttpSecurity — used to specify rules to restrict access of URLs or URL patterns, Security Filter class(es), etc...

UserDetails, UserDetailsService — UserDetails stores details of currently logged in User. UserDetailsService provides us UserDetails

RestTemplate — to consume REST API end points exposed by other Microservices of the Application or external sources. RestTemplate is blocking in nature, use **WebClient** to develop reactive, non blocking client.

DataSource — holds details of a data source such as a database, with which application interacts

MockMvc — MockMvc is a utility class that lets us send mock HTTP requests, for Testing REST API end points.

There are many more classes and interfaces provided by Spring & Spring Boot, in this post I have mentioned most commonly used classes and interfaces, only.

I will be adding few important code snippets showing usage of above classes or interfaces, in second part of this post, shortly.

My other Blog Posts:

Spring, Spring Boot Annotations Cheat sheet

As known there are a number of Annotations provided by Java's Spring, Spring Boot Framework, and it would be quite...

medium.com

<div><div><div><div><div><div>How to create Custom Java Collection by extending from an existing Collection class</div><div>Though Core Java provides a number of builtin Collections such as ArrayList, LinkedList, HashSet, etc...., sometimes it...</div><div>medium.com</div></div></div></div></div></div>	

https://medium.com/@sureshkumar_95502/angular-cheat-sheet-b542fc75ddf

- Spring Boot Starter

Spring Boot Classes

Spring Boot Interfaces
- Spring Boot Cheat Sheet

Spring Cheat Sheet

[About](#) [Help](#) [Terms](#) [Privacy](#)

Get the Medium app

