Spring and Spring Boot - Basic Notes (01-07-2025)

* Github Repository of code of 01-07-2025:

<https://github.com/Kashish-HUB/Product_Spring>

# 🌱 Spring Framework – Basics

Spring is a lightweight, open-source Java framework for building enterprise applications.

## 🔧 Key Features

- Dependency Injection (DI): Loose coupling between components.  
- Aspect-Oriented Programming (AOP): Separate cross-cutting concerns like logging, security.  
- Transaction Management: Handles database transactions.  
- Spring MVC: Framework for building web applications.  
- Integration Support: JDBC, JPA, JMS, REST, etc.

## 🧱 Spring Modules

- Core Container: BeanFactory, ApplicationContext  
- Spring AOP  
- Spring Data Access: JDBC, ORM  
- Spring Web: Spring MVC  
- Spring Security

# 🚀 Spring Boot – Basics

Spring Boot is a project on top of Spring that simplifies Spring application development.

## 🎯 Goal

Create stand-alone, production-grade Spring apps quickly with minimal configuration.

## ⚙️ Features

- Auto-Configuration: Automatically configures your app based on dependencies.  
- Embedded Servers: Tomcat, Jetty, or Undertow built-in.  
- No XML Configuration: Uses annotations & application.properties/yml.  
- Starter Dependencies: Predefined dependency sets (e.g., spring-boot-starter-web).  
- Actuator: Provides production-ready endpoints (health, metrics, etc.)

# 📌 Common Spring Annotations

|  |  |
| --- | --- |
| Annotation | Use |
| @Component | Generic Spring-managed component |
| @Service | Marks a service class |
| @Repository | Marks a DAO class, adds DB exception translation |
| @Controller | Web controller in Spring MVC |
| @RestController | Combines @Controller + @ResponseBody |
| @Autowired | Injects dependencies automatically |
| @Configuration | Class with Spring bean definitions |
| @Bean | Declares a bean in @Configuration class |
| @RequestMapping | Maps HTTP requests to handler methods |

# 🛠️ Spring Boot Annotations

|  |  |
| --- | --- |
| Annotation | Use |
| @SpringBootApplication | Main entry point – combines @Configuration, @EnableAutoConfiguration, @ComponentScan |
| @EnableAutoConfiguration | Enables Spring Boot auto-configuration |
| @ComponentScan | Tells Spring where to look for components |
| @Value | Injects values from application.properties |

# 📂 Project Structure (Spring Boot)

src/  
 └── main/  
 ├── java/  
 │ └── com.example.app/  
 │ ├── controller/  
 │ ├── service/  
 │ ├── repository/  
 │ └── App.java <-- Main class  
 └── resources/  
 ├── application.properties  
 └── templates/

# 📝 application.properties (Example)

server.port=8081  
spring.datasource.url=jdbc:mysql://localhost:3306/mydb  
spring.datasource.username=root  
spring.datasource.password=1234  
spring.jpa.hibernate.ddl-auto=update

# 🚀 Running a Spring Boot App

# Using Maven  
mvn spring-boot:run  
  
# OR build JAR and run  
mvn clean install  
java -jar target/app-name.jar