Parul® University

NAAC A++

Enterprise Programming using JAVA

Chapter-5: Spring

Prof. ARNIKA PATEL

Assistant Professor

Department of CSE

Content

INDEX

Spring

- Spring Framework is a comprehensive and versatile platform for enterprise Java development. It is known for its **Inversion of Control (IoC)** and **Dependency Injection (DI)** capabilities that simplify creating modular and testable applications.

- Key features include **Spring MVC** for web development, **Spring Boot** for rapid application setup, and **Spring Security** for robust authentication and authorization.

Spring

- With a rich ecosystem covering **Spring Data** for database interactions and **Spring Cloud** for building microservices, **Spring** supports scalable and resilient enterprise solutions, making it an essential framework for developers of all experience levels.

- Spring framework is a Java platform that is open source. Rod Johnson created it, and it was first released under the Apache 2.0 license in June 2003. When it comes to size and transparency, Spring is a featherweight. Spring framework's basic version is about 2MB in size.

Spring

Applications of Spring

- **POJO Based:** Spring allows developers to use POJOs to create enterprise-class apps
- **Modular** : Spring is set up in a modular approach.
- **Integration with Existing Frameworks:** Spring does not reinvent the wheel; rather, it makes extensive use of existing technologies such as numerous ORM frameworks, logging frameworks, JEE, Quartz, and JDK timers, and other view technologies.
- **Testability:** Because environment-dependent code is put into this framework, testing a Spring-written application is trivial.

Spring

Applications of Spring

- **Web MVC:** Spring's web framework is a well-designed web MVC framework that is an excellent alternative to web frameworks like Struts and other over-engineered or less popular web frameworks.
- **Central Exception Handling:** Spring provides a handy API for converting technology-specific exceptions (such as those raised by JDBC, Hibernate, or JDO) into consistent, unchecked exceptions.
- **Lightweight:** IoC containers are typically lightweight, especially when compared to EJB containers, for example.

Spring Architecture

- The **Spring framework** is a widely used open-source Java framework that provides a comprehensive programming and configuration model for building enterprise applications.

- Its architecture is designed around two core principles:
  1. Dependency Injection (DI)
  2. Aspect-Oriented Programming (AOP)

Spring Architecture

1. **Dependency Injection (DI)**

- DI is a design pattern that promotes loose coupling between components by allowing objects to be injected into a class rather than the class creating them itself. The Spring framework provides an Inversion of Control (IoC) container which is responsible for managing these dependencies.
- The BeanFactory and ApplicationContext are the core interfaces of the IoC container.
  - BeanFactory provides basic dependency injection.
  - ApplicationContext adds advanced features like internationalization, event propagation, and resource loading.
- Spring manages the lifecycle of beans, including initialization, destruction, and scopes (e.g., singleton, prototype, request, session).

Spring Architecture

2. Aspect-Oriented Programming (AOP)

- AOP allows developers to modularize cross-cutting concerns such as logging, security, and transaction management.

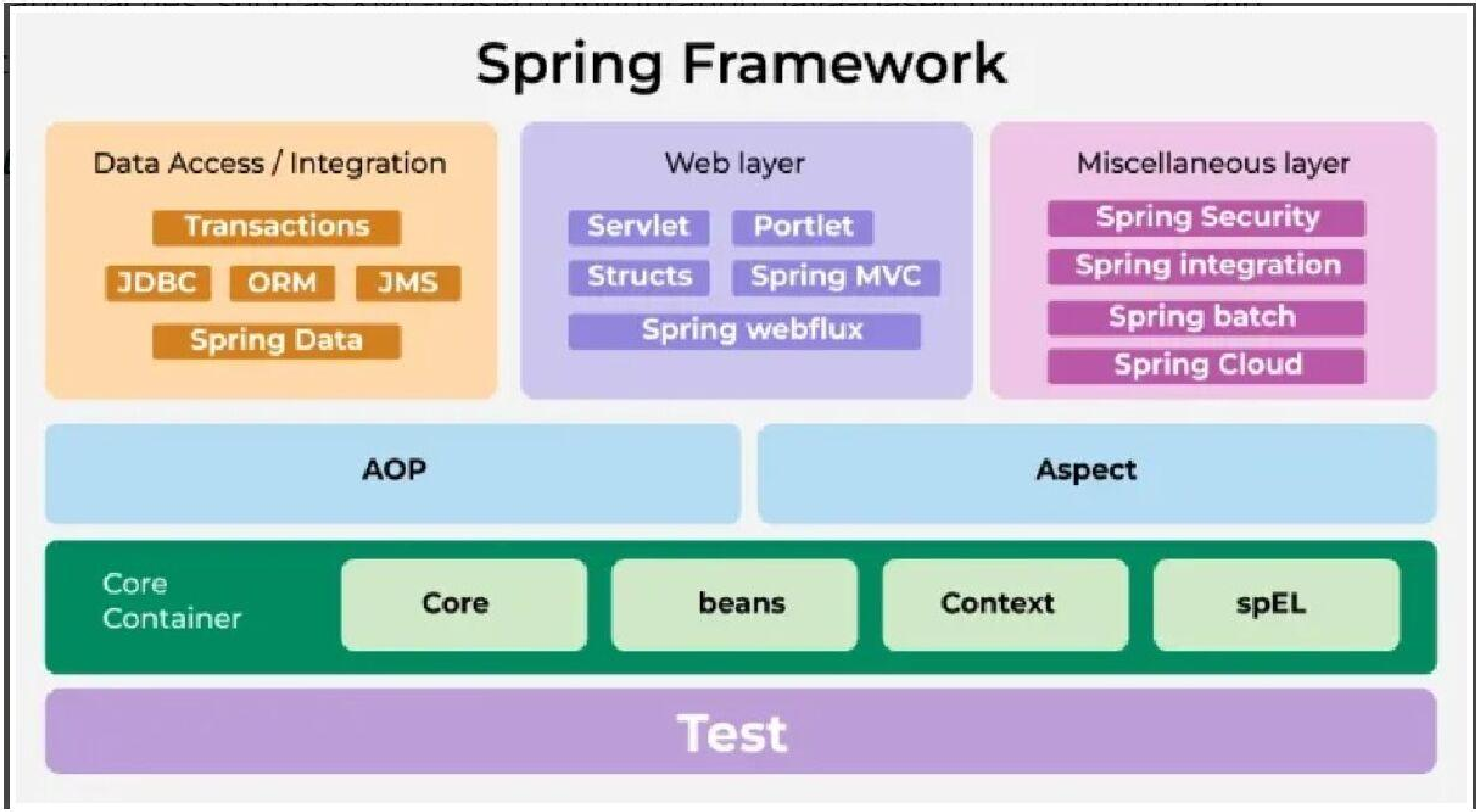
- These concerns can be applied across multiple components without modifying the core business logic.

- Spring AOP uses advice, pointcuts, and aspects to implement cross-cutting concerns. For example, you can use AOP to log method execution times or enforce security checks

Spring Architecture

- Overall, the Spring framework architecture is based on the principles of modularity, separation of concerns, and flexibility, providing developers with a powerful set of tools to build robust, scalable, and maintainable enterprise applications.

Spring Modules



Spring Modules

The Spring framework consists of several **modules**, which can be categorized into four main areas:

1. **Core Container:** The Core Container provides the fundamental functionality of the Spring framework, including the IoC container and ApplicationContext.
2. **Data Access/Integration:** The Data Access/Integration area provides support for integrating with databases and other data sources.
3. **Web:** The Web area provides support for building web applications, including the Spring MVC and Spring WebFlux modules.
4. **Miscellaneous:** The Miscellaneous area includes other modules that provide additional functionality, such as the Spring Security module for authentication and authorization features.

Spring Modules

Core Container

- **Spring Core:** This module provides the fundamental functionality of the Spring framework, including IoC and DI. The IoC container is the heart of the Spring Framework, responsible for creating and managing instances of JavaBeans. It uses dependency injection to wire the beans together.
- **Spring Beans:** This module provides the BeanFactory, which is the basic building block of the IoC container, and the BeanWrapper, which is responsible for managing the lifecycle of a bean. The Bean Factory is the core interface for accessing the IoC container. It provides methods for retrieving beans.

Spring Modules

Core Container

- **Spring Context:** This module provides the ApplicationContext, which is an advanced version of the BeanFactory and provides additional features, such as internationalization and resource loading, and the ability to publish and consume events.
- **Spring Expression Language (SpEL):** This module provides a powerful expression language for querying and manipulating objects during runtime. SpEL supports a wide range of features, including property access, method invocation, conditionals, loops, and type conversion.

Spring Modules

Data Access/Integration:

- **Spring ORM:** Spring ORM provides a higher-level abstraction layer on top of ORM frameworks, allowing developers to write less boilerplate code and more easily integrate ORM technologies with other Spring features, such as transaction management and caching.

- **Spring Data:** This module provides a consistent and easy-to-use programming model for working with data access technologies, including databases, NoSQL, and cloud-based data services.

Spring Modules

Data Access/Integration:

- **Spring Transaction:** This module provides support for declarative transaction management in Spring applications. Spring Transaction provides support for various transaction propagation and isolation levels, allowing developers to manage transactions at different levels of granularity.

Spring Modules

Web

- **Spring MVC:** This module provides a Model-View-Controller (MVC) framework for building web applications. Spring MVC provides a range of features, including support for handling HTTP requests and responses, form handling, data binding, validation, and more.
- **Spring WebFlux:** This module provides a reactive programming model for building web applications that require high concurrency and scalability.
- **Spring Web Services:** This module provides support for building SOAP-based and RESTful web services. Spring Web Services provides support for generating WSDL (Web Services Description Language) from Java classes, and for generating Java classes from WSDL.

Spring Modules

Miscellaneous

- **Spring Security:** This module provides authentication and authorization features for Spring applications. Spring Security provides a range of authorization mechanisms, such as role-based access control and expression-based access control.
- **Spring Integration:** This module provides support for building message-driven and event-driven architectures.
- **Spring Batch:** This module provides support for batch processing and integration with enterprise systems.
- **Spring Cloud:** This module provides support for building cloud-native applications using Spring technologies.

Parul® University NAAC A++

PPT Content Resources Reference Sample:

1. **Book Reference**

   Jim Farley, William Crawford, David Flanagan. Java Enterprise in a Nutshell, O'Reilly

2. **Book Reference**

   Rocha, R., Purificação, J. (2018). Java EE 8 Design Patterns and Best Practices: Build Enterprise-ready Scalable Applications with Architectural Design Patterns. Germany: Packt Publishing..

3. **Website Reference**

   https://www.scribd.com/document/268349254/Java-8-Programming-Black-Book .

4. **Sources**

   https://developers.redhat.com/topics/enterprise-java

5. **Article**

   https://www.researchgate.net/publication/276412369_Advanced_Java_Programming

Parul® University | NAAC GRADE A++

https://paruluniversity.ac.in/