



Spring, SpringBoot, JPA, Hibernate

A hand-drawn rectangular box with a blue outline and a hatched bottom edge. The text "Spring, SpringBoot, JPA, Hibernate" is written in a bold, dark blue font inside the box. The background is white and decorated with various colorful, hand-drawn shapes: a green squiggle in the top left, a yellow checkmark in the top center, a blue arrow pointing up in the top right, a yellow square and blue squiggle on the middle left, a yellow circle and red triangle on the middle right, a green triangle in the bottom left, a blue triangle in the bottom center, and a red triangle in the bottom right.


Agenda of the course




Intro to Spring framework

- *What is Spring?*
- *Spring Vs Java EE*
- *Evolution of Spring*
- *Spring release timeline*
- *Different projects inside Spring*


Spring Core

- *Inversion of Control (IoC)*
 - *Dependency Injection (DI)*
 - *Different approaches for Beans creation*
 - *Autowiring, Bean Scopes*
 - *Aspect-Oriented Programming (AOP)*
- 

Spring MVC

- 
- *Introduction to MVC pattern*
 - *Overview of Web Apps*
 - *How Web Apps works ?*
 - *Spring MVC internal flow*
 - *Create a web App using Spring MVC*
 - *Spring MVC validations*

Agenda of the course




Thymeleaf

- *How to build dynamic web apps using Thymeleaf & Spring*
- *Thymeleaf integration with Spring, Spring MVC, Spring Security*

Spring Boot

- *Why Spring Boot?*
 - *Auto-configuration*
 - *Build Web Apps using SpringBoot, Thymeleaf, Spring MVC*
 - *Spring Boot Dev Tools*
 - *Spring Boot H2 Database*
 - *Connecting to AWS MYSQL DB*
- 

Spring Security



- *Authentication & Authorization*
- *Securing web Apps/REST APIs using Spring Security*
- *Role based access*
- *Cross-Site Request Forgery (CSRF)*
- *Cross-Origin Resource Sharing (CORS)*


Agenda of the course



Spring JDBC

- *Problems with Core Java JDBC*
- *Advantages with Spring JDBC*
- *Performing CRUD operations with Spring JDBC*
- *Details about JdbcTemplate*
- *RowMapper*

Spring Data

- *Why do we need ORM frameworks?*
 - *Introduction to JPA*
 - *Derived Query methods in JPA*
 - *OneToOne, OneToMany, ManyToOne, ManyToMany mappings inside JPA/Hibernate*
 - *Sorting, Pagination, JPQL*
- 

Spring REST

- 
- *Building Rest Services*
 - *Consuming Rest Services using OpenFeign, Web Client, RestTemplate*
 - *Securing Rest Services*
 - *Spring Data Rest*
 - *HAL Explorer*


Agenda of the course




Logging

- *Logging severities*
- *How to make logging configurations inside Spring Boot*
- *Writing logs into a file*

Properties & Profiles

- *How to define custom properties*
 - *How to read properties in Java code*
 - *Deep dive on Spring Profiles*
 - *Activating a Profile*
 - *Conditional Bean creation using Profile*
- 

Actuator

- 
- *Introduction to SpringBoot Actuator*
 - *Exploring the APIs of Actuator*
 - *Securing Actuator*
 - *Viewing Actuator Data in Spring Boot Admin*

Agenda of the course



**DEPLOYING
SPRINGBOOT
WEBAPP INTO
CLOUD (AWS)**





The Spring Framework (shortly, Spring) is a mature, powerful and highly flexible framework focused on building web applications in Java.

Spring makes programming Java quicker, easier, and safer for everybody. It's focus on speed, simplicity, and productivity has made it the world's most popular Java framework.

Whether you're building secure, reactive, cloud-based microservices for the web, or complex streaming data flows for the enterprise, Spring has the tools to help.

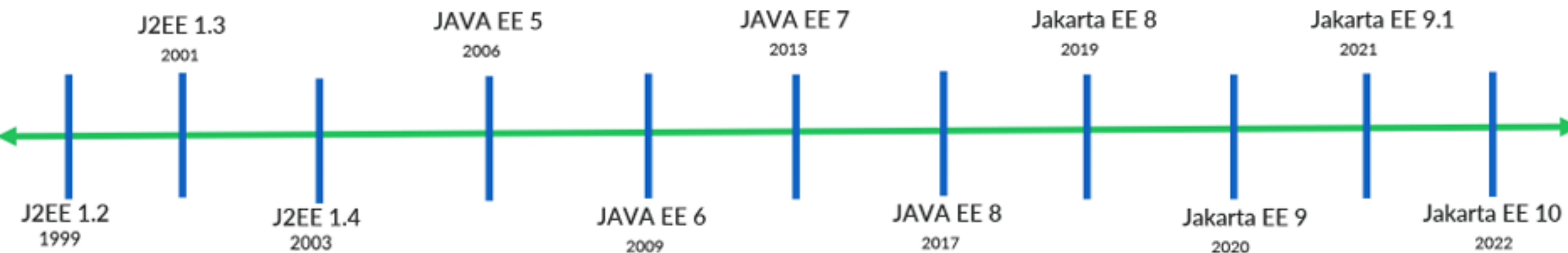
Born as an alternative to EJBs in the early 2000s, the Spring framework quickly overtook its opponent with its simplicity, variety of features, and its third-party library integrations.

It is so popular, that its main competitor quit the race when Oracle stopped the evolution of Java EE 8, and the community took over its maintenance via Jakarta EE.

The main reason of Spring framework success is, it regularly introduces features/projects based on the latest market trends, needs of the Dev community. For ex: SpringBoot

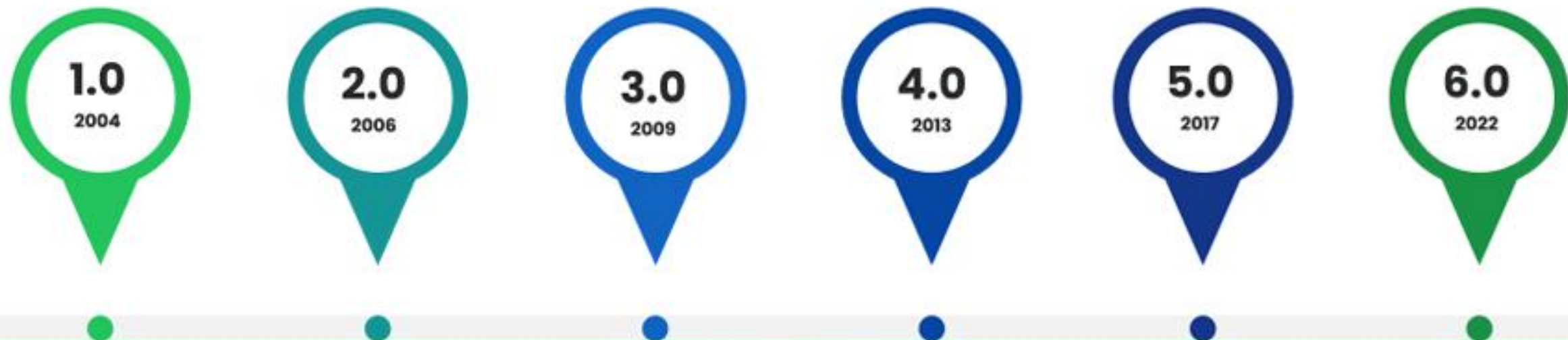
Spring is open source. It has a large and active community that provides continuous feedback based on a diverse range of real-world use cases.

JAVA EE RELEASE TIMELINE



- *Java/Jakarta Enterprise Edition (EE) contains Servlets, JSPs, EJB, JMS, RMI, JPA, JSF, JAXB, JAX-WS, Web Sockets etc.*
- *Components of Java/Jakarta Enterprise Edition (EE) like EJB, Servlets are complex in nature due to which everyone adapted Spring framework for web applications development.*
- *Java EE quit the race against the Spring framework, when Oracle stopped the evolution of Java EE 8, and the community took over its maintenance via Jakarta EE.*
- *Since Oracle owns the trademark for the name "Java", Java EE renamed to Jakarta EE. All the packages are updated with javax.* to jakarta.* namespace change.*

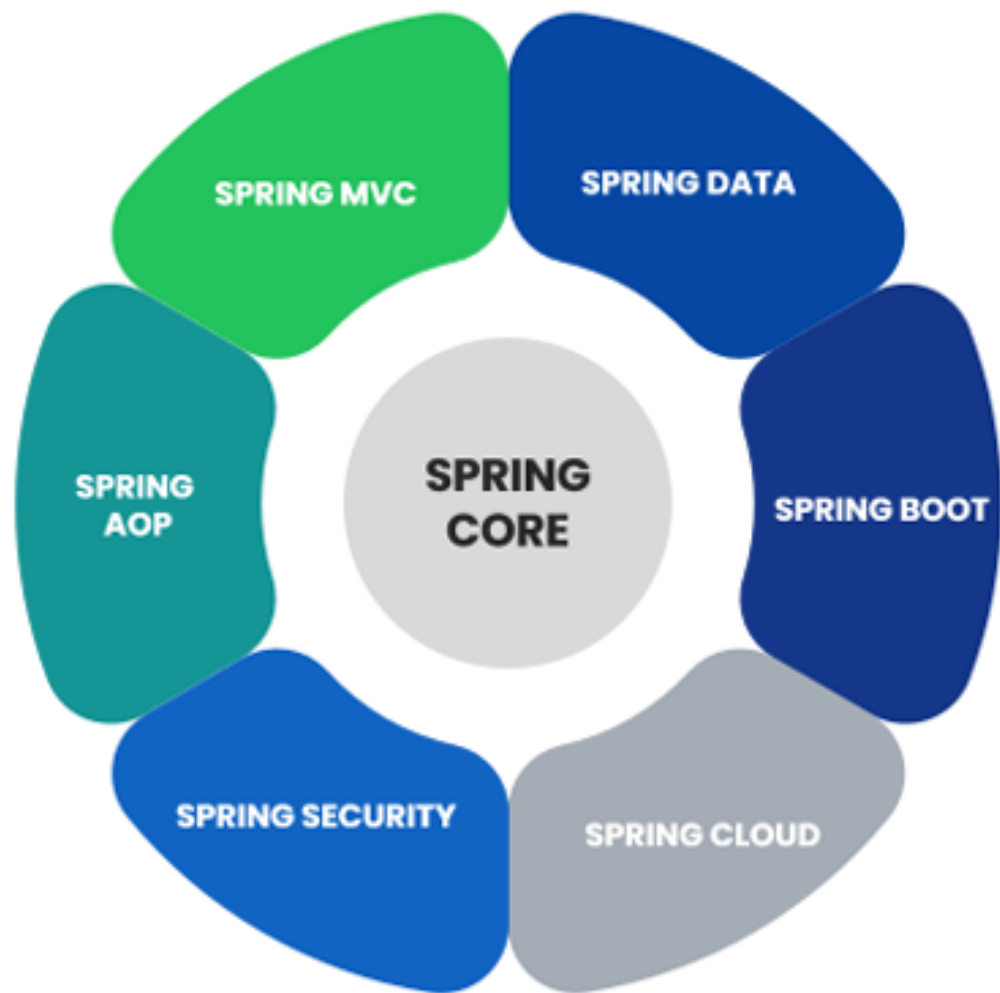
SPRING RELEASE TIMELINE



SPRING VERSION HISTORY

- The first version of Spring was written by Rod Johnson, who released the framework with the publication of his book *Expert One-on-One J2EE Design and Development* in October 2002
- Spring came into being in 2003 as a response to the complexity of the early J2EE specifications. While some consider Java EE and Spring to be in competition, Spring is, in fact, complementary to Java EE. The Spring programming model does not embrace the Java EE platform specification; rather, it integrates with carefully selected individual specifications from the EE umbrella
- Spring continues to innovate and to evolve. Beyond the Spring Framework, there are other projects, such as Spring Boot, Spring Security, Spring Data, Spring Cloud, Spring Batch, among others.

SPRING CORE



- Spring Core is the heart of entire Spring. It contains some base framework classes, principles and mechanisms.
- The entire Spring Framework and other projects of Spring are developed on top of the Spring Core.
- Spring Core contains following important components,
 - ✓ IoC (Inversion of Control)
 - ✓ DI (Dependency Injection)
 - ✓ Beans
 - ✓ Context
 - ✓ SpEL (Spring Expression Language)
 - ✓ IoC Container


INVERSION OF CONTROL & DEPENDENCY INJECTION



**CORE PRINCIPLES
OF SPRING**

- Inversion of Control (IoC) is a Software Design Principle, independent of language, which does not actually create the objects but describes the way in which object is being created.
- IoC is the principle, where the control flow of a program is inverted: instead of the programmer controlling the flow of a program, the framework or service takes control of the program flow.
- Dependency Injection is the pattern through which Inversion of Control achieved.
- Through Dependency Injection, the responsibility of creating objects is shifted from the application to the Spring IoC container. It reduces coupling between multiple objects as it is dynamically injected by the framework.


ADVANTAGES OF IoC & DI




LOOSE COUPLING
BETWEEN THE
COMPONENTS




MINIMIZES THE
AMOUNT
OF CODE IN YOUR
APPLICATION



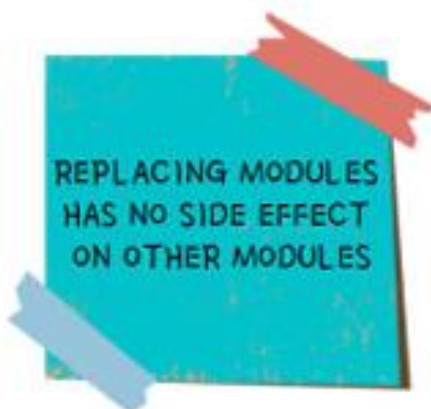
MAKES UNIT
TESTING EASY WITH
DIFFERENT MOCKS



INCREASED SYSTEM
MAINTAINABILITY &
MODULE REUSABILITY



ALLOWS CONCURRENT
OR INDEPENDENT
DEVELOPMENT



REPLACING MODULES
HAS NO SIDE EFFECT
ON OTHER MODULES



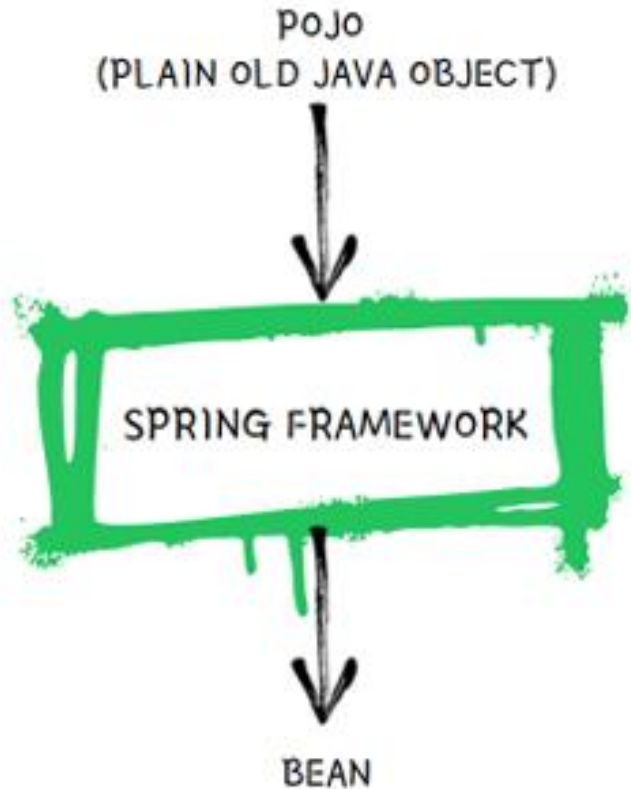
**TIGHT COUPLING
SCENARIO**



**LOOSE COUPLING
SCENARIO**

**REAL LIFE
LOOSE
COUPLING
EXAMPLE**

SPRING BEANS, CONTEXT, SpEL



- Any normal Java class that is instantiated, assembled, and otherwise managed by a Spring IoC container is called Spring Bean.
- These beans are created with the configuration metadata that you supply to the container either in the form of XML configs and Annotations.
- Spring IoC Container manages the lifecycle of Spring Bean scope and injecting any required dependencies in the bean.
- Context is like a memory location of your app in which we add all the object instances that we want the framework to manage. By default, Spring doesn't know any of the objects you define in your application. To enable Spring to see your objects, you need to add them to the context.
- The SpEL provides a powerful expression language for querying and manipulating an object graph at runtime like setting and getting property values, property assignment, method invocation etc.

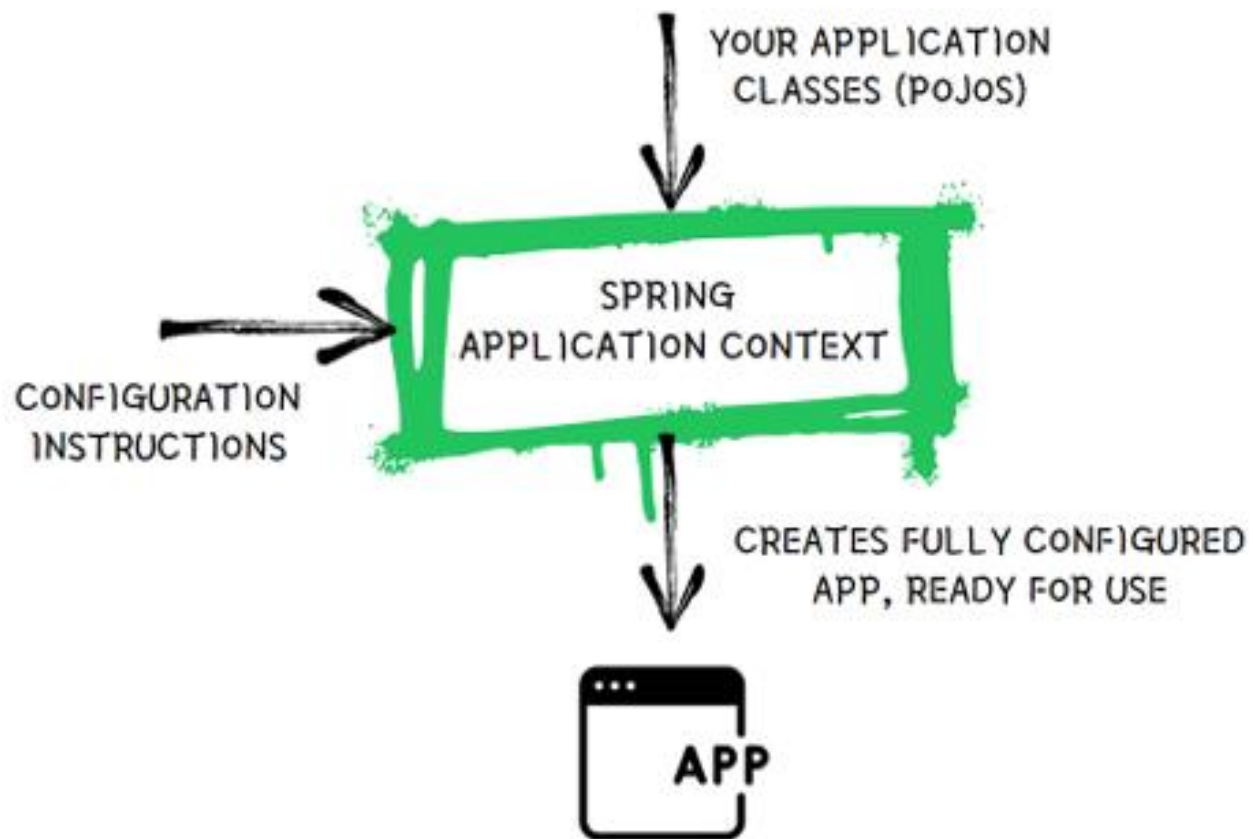
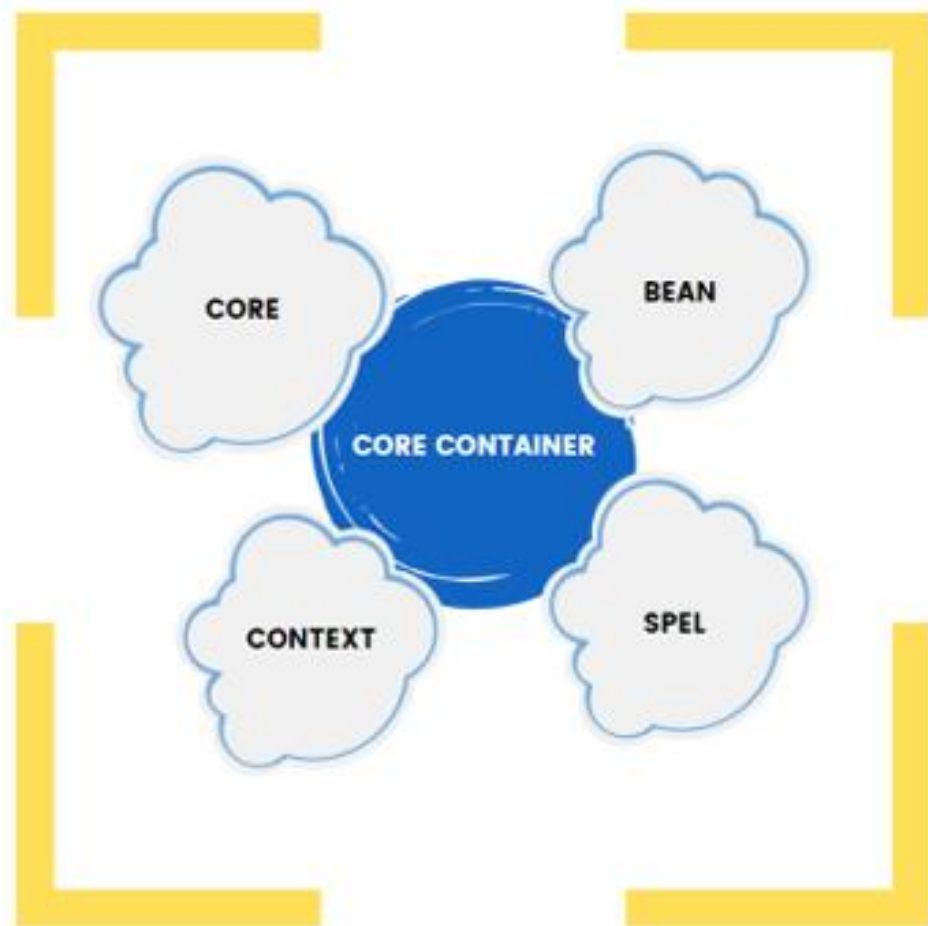
SPRING IoC CONTAINER



Spring IoC Container

- The IoC container is responsible
 - ✓ to instantiate the application class
 - ✓ to configure the object
 - ✓ to assemble the dependencies between the objects
- There are two types of IoC containers. They are:
 - ✓ `org.springframework.beans.factory.BeanFactory`
 - ✓ `org.springframework.context.ApplicationContext`
- The Spring container uses dependency injection (DI) to manage the components/objects that make up an application.

SPRING IoC CONTAINER



MAVEN

