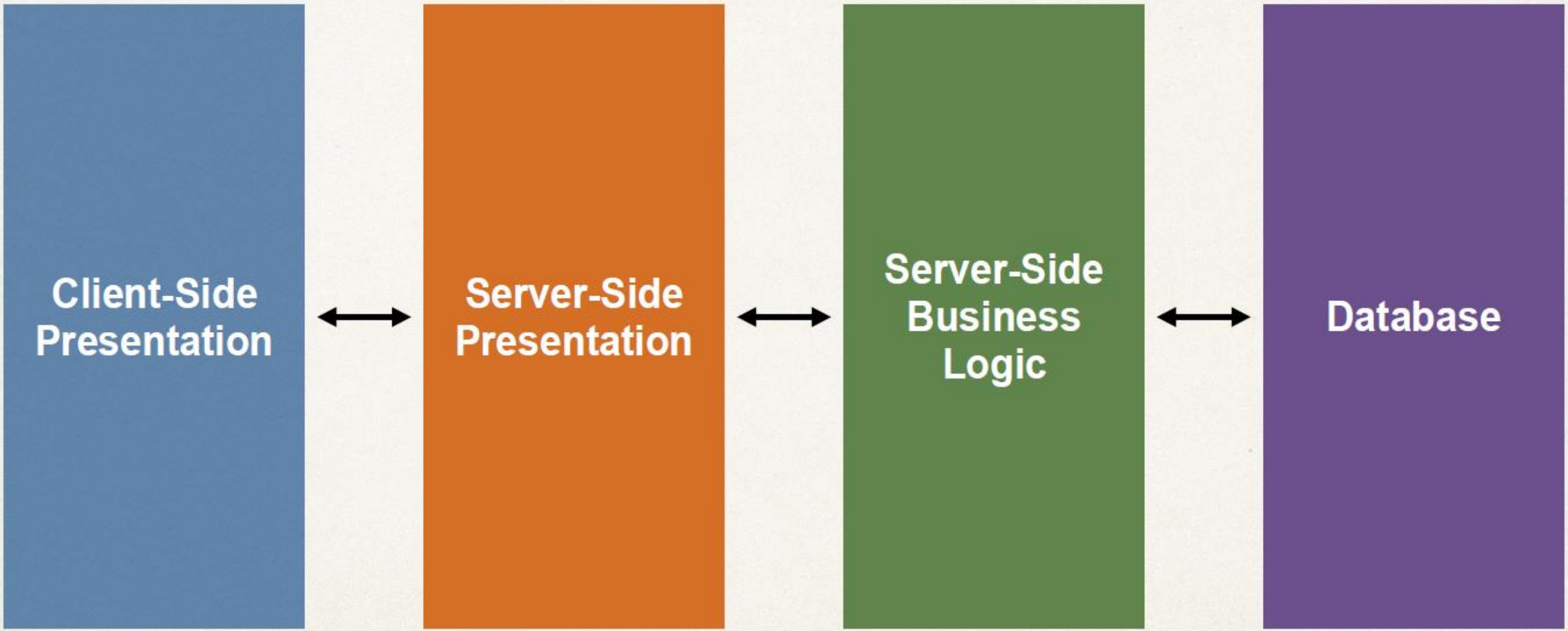
Why Spring?

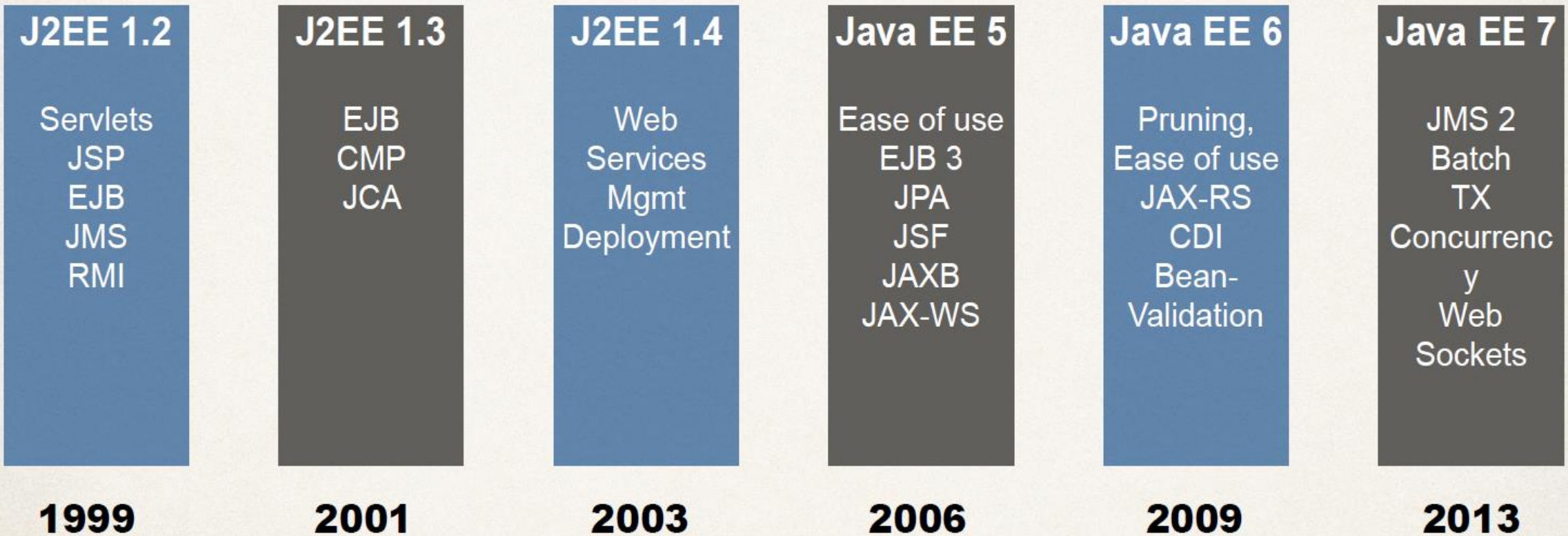
Spring in a Nutshell

- Very popular framework for building Java applications
- Initially a simpler and lightweight alternative to J2EE
- Provides a large number of helper classes … makes things easier

But What About J2EE???



But What About Java EE???



EJB v1 and v2 - Complexity

- Early version of EJB (v1 and v2) were extremely complex!!!
- Multiple deployment descriptors
- Multiple interfaces
- Poor Performance of Entity Beans

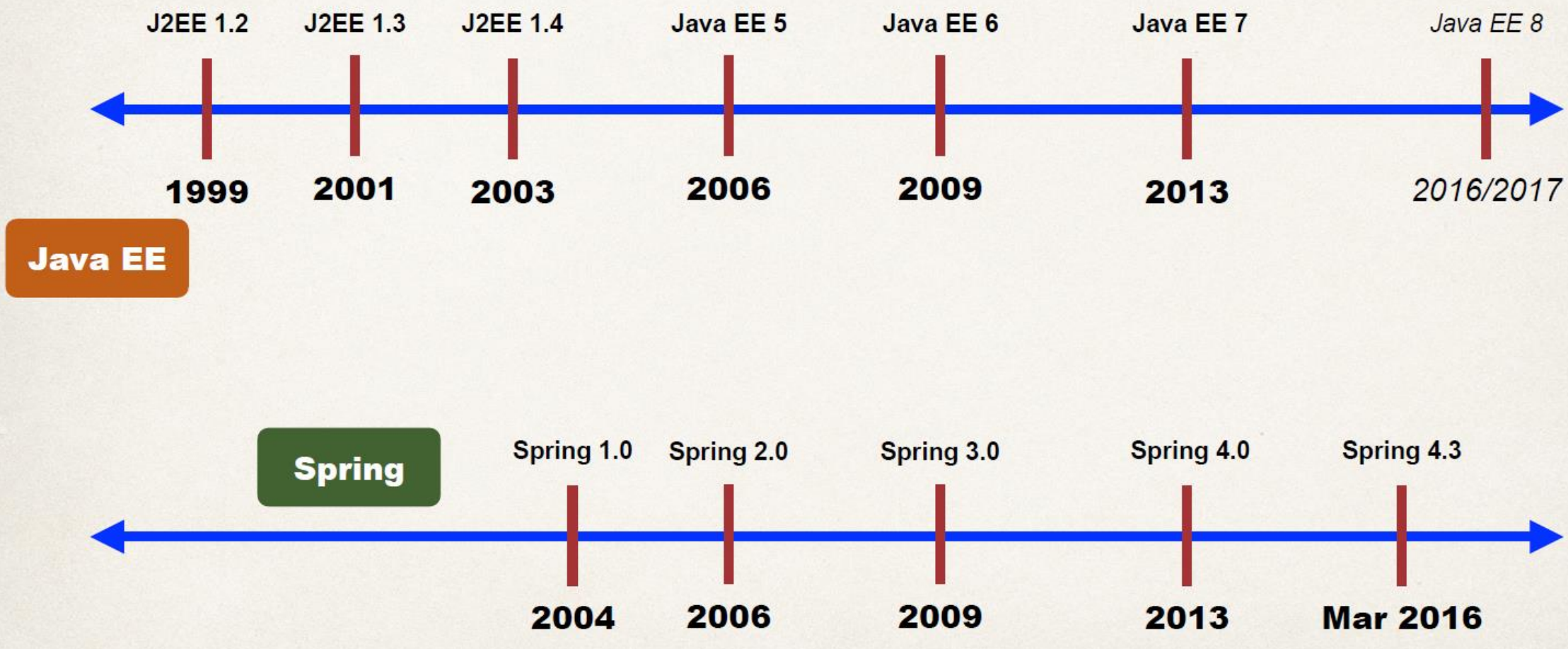
EJB Client

Home Interface
Home Object

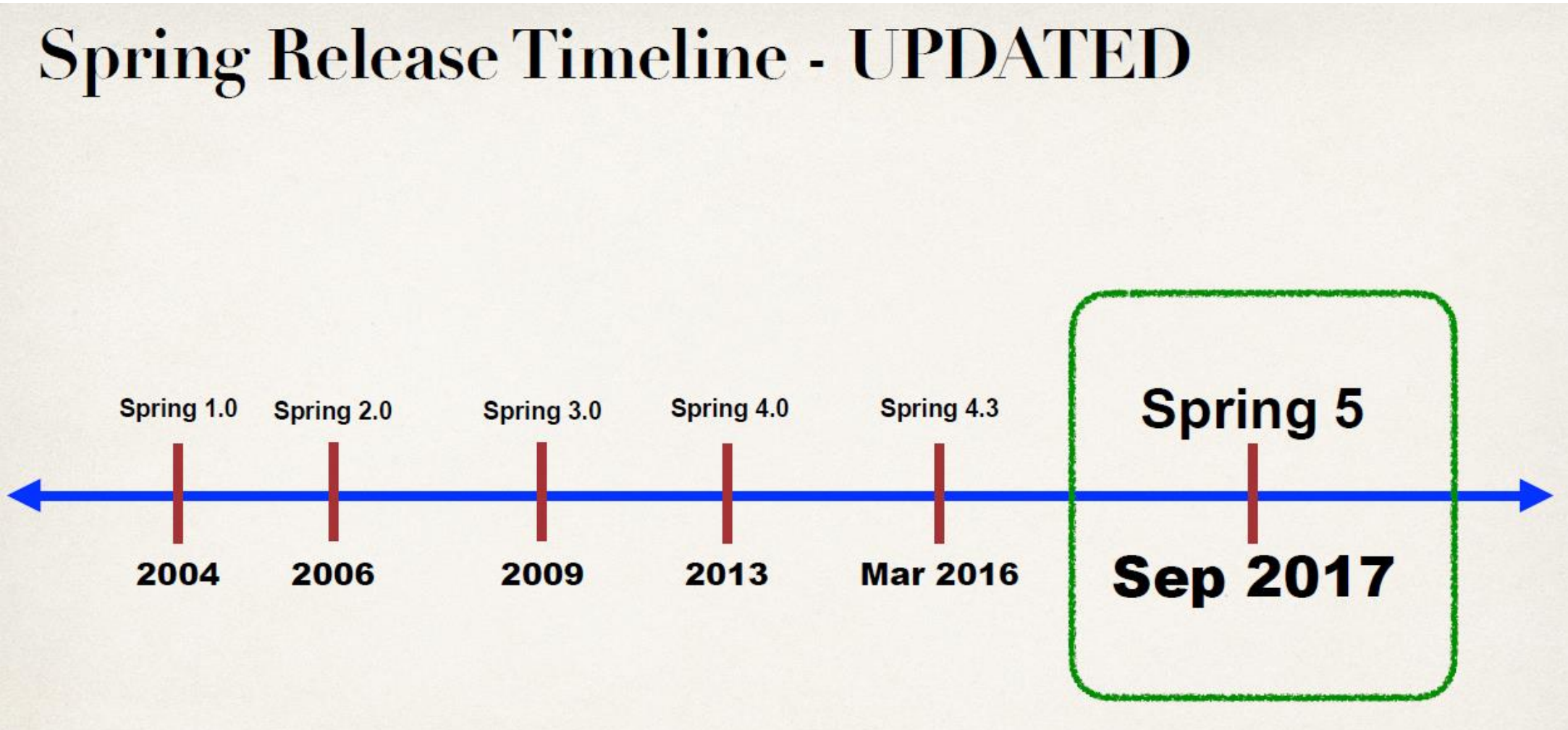
Component Interface
EJB Object

Bean Class

Release Timeline



Spring Release Timeline - UPDATED



What's New in Spring 5

- Updated minimum requirements for Java 8 or higher
- Deprecated legacy integration for: Tiles, Velocity, Portlet, Guava etc
- Upgraded Spring MVC to use new versions of Servlet API 4.0
- Added new reactive programming framework: Spring WebFlux

Spring Framework Overview

Spring Website - Official

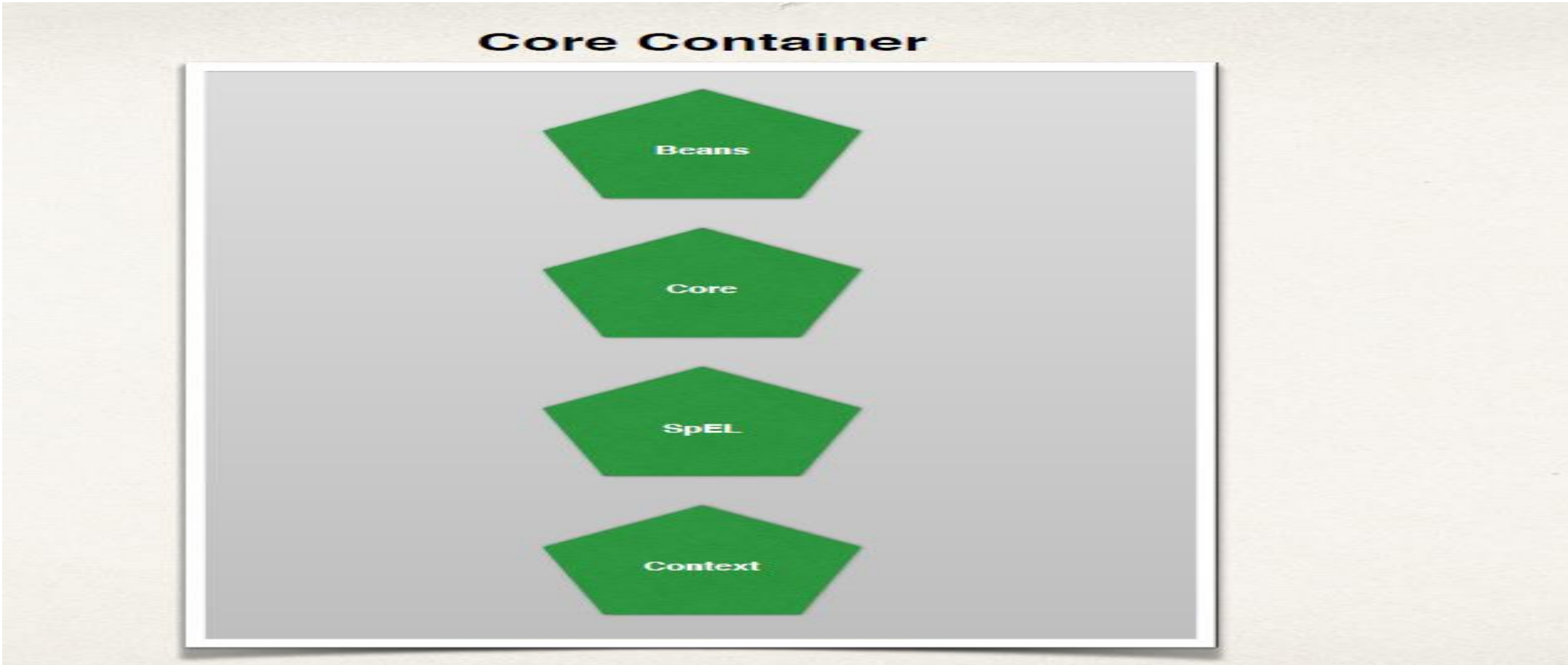
www.spring.io

Why Spring?

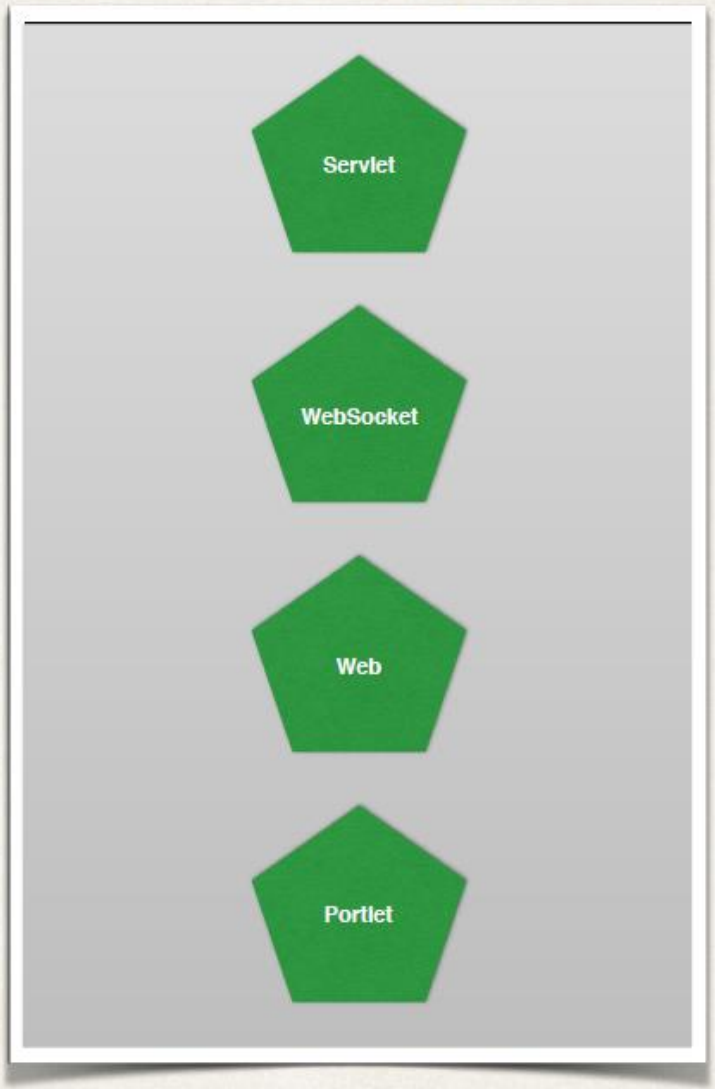
Simplify Java Enterprise Development

Goals of Spring

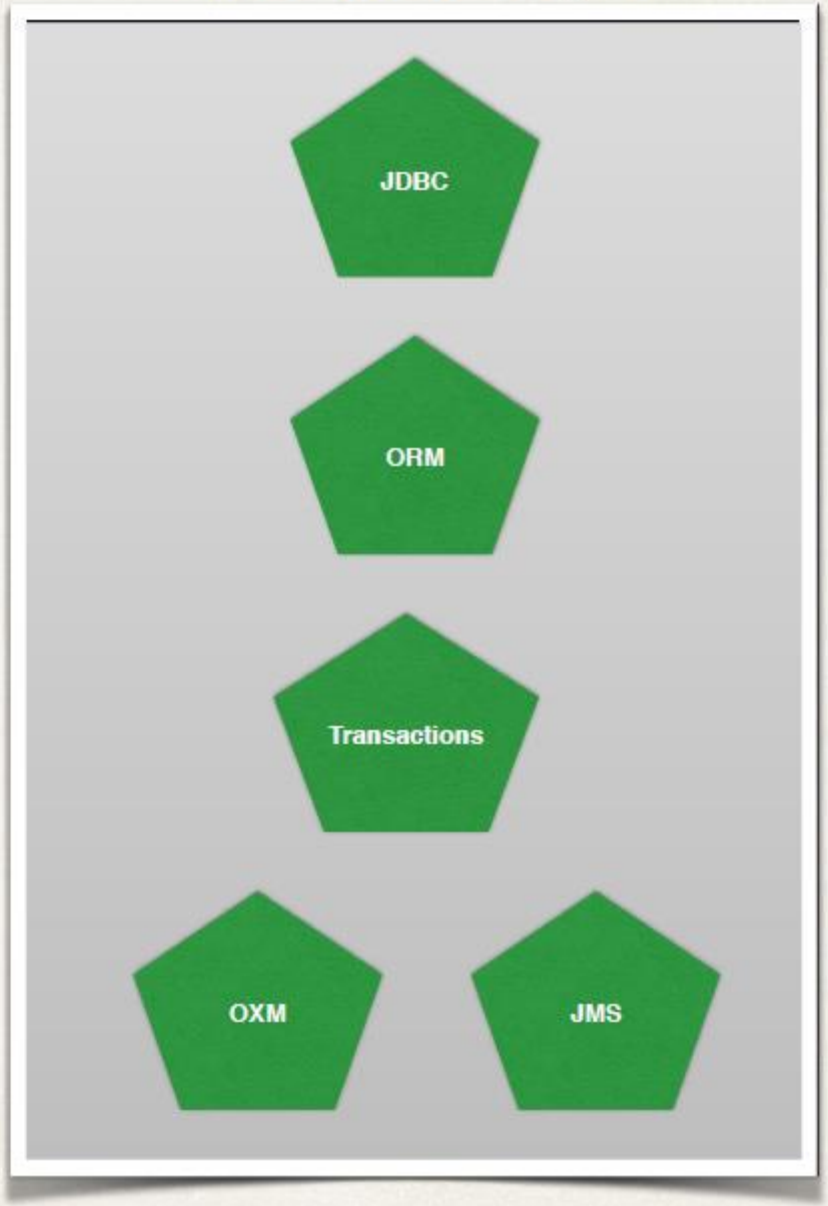
- Lightweight development with Java POJOs (Plain-Old-Java-Objects)
- Dependency injection to promote loose coupling
- Declarative programming with Aspect-Oriented-Programming (AOP)
- Minimize boilerplate Java code


Core Container
Beans
Core
SpEL
Context

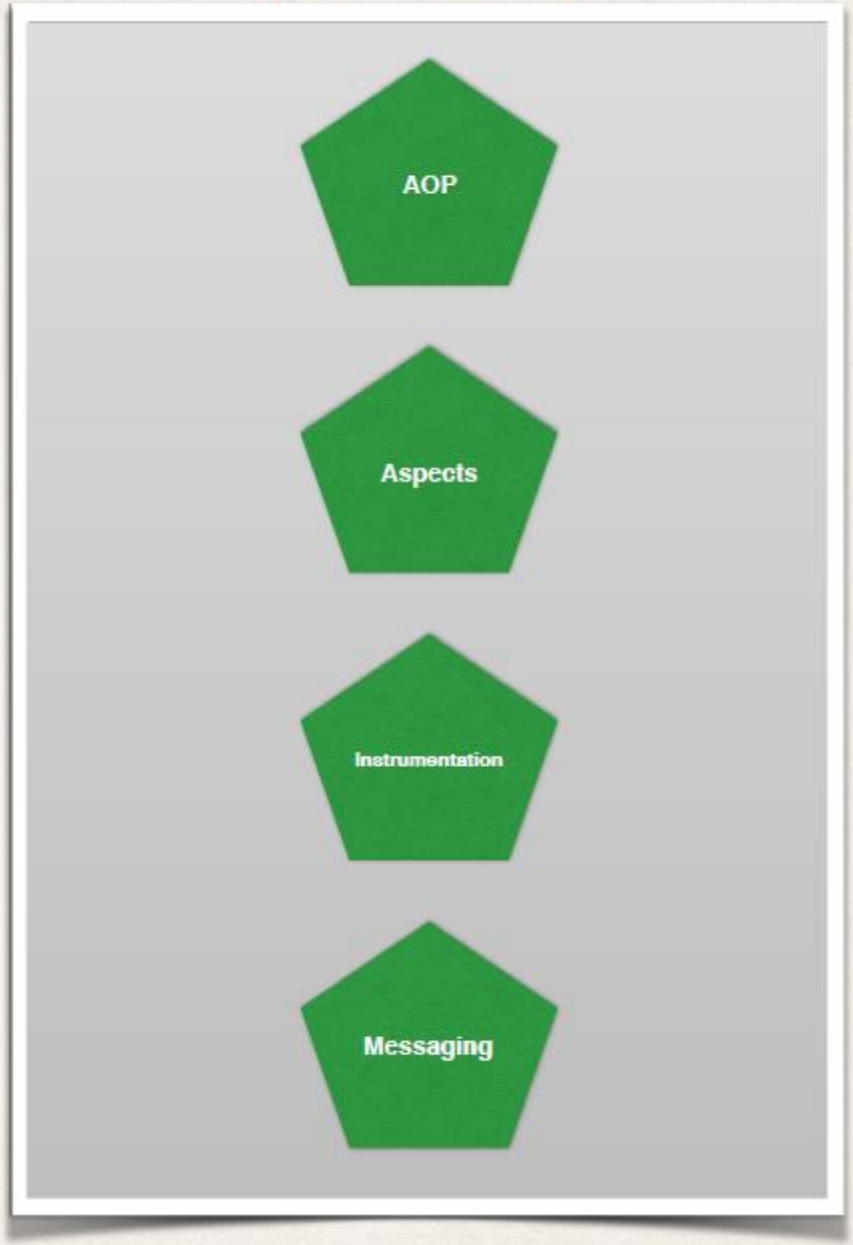
Web Layer



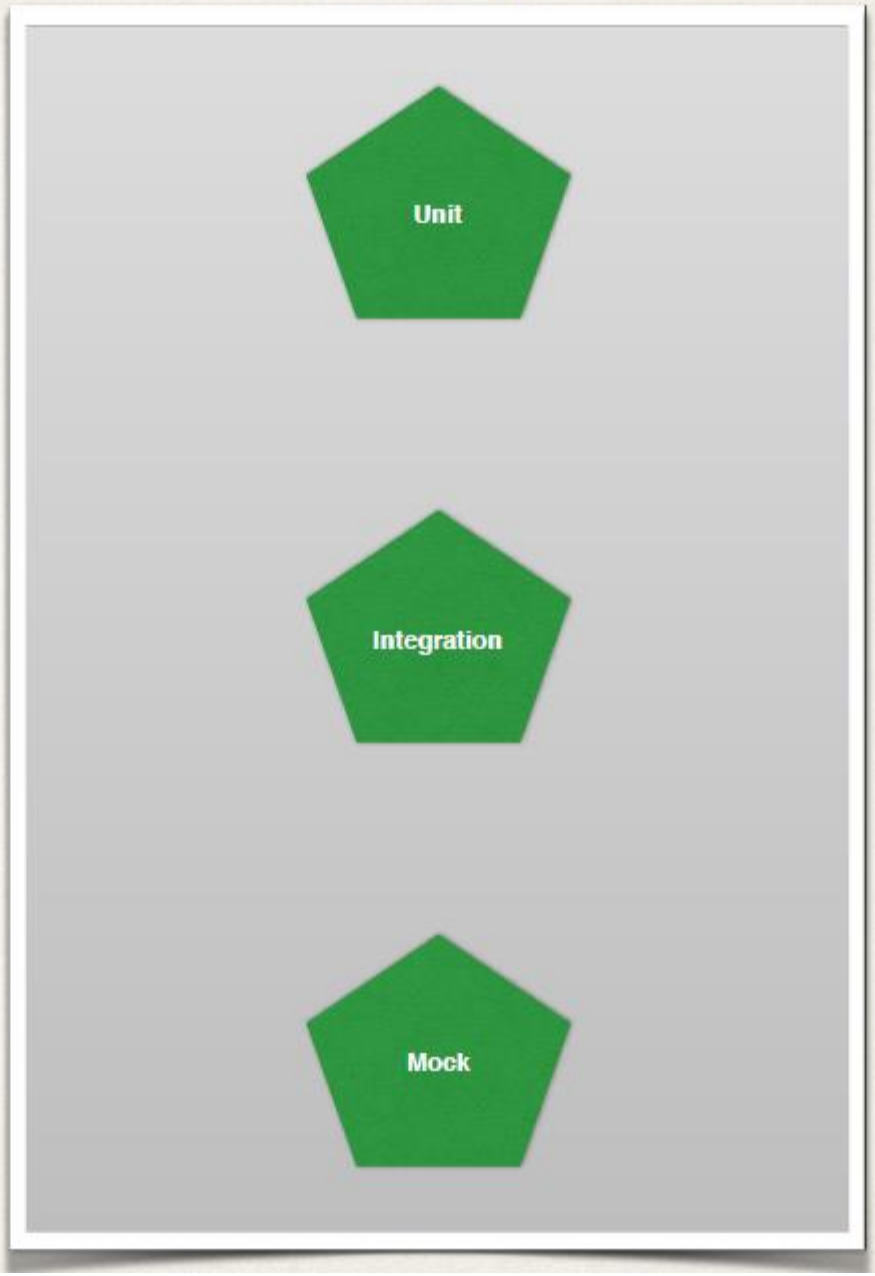
Data Access Layer



Infrastructure



Test Layer



Spring Projects

Spring Projects

What Are Spring "Projects"

- Additional Spring ***modules*** built-on top of the core Spring Framework
- Only use what you need …
  - Spring Cloud, Spring Data
  - Spring Batch, Spring Security
  - Spring for Android, Spring Web Flow
  - Spring Web Services, Spring LDAP

What Are Spring "Projects"

- Additional Spring ***modules*** built-on top of the core Spring Framework
- Only use what you need …
  - Spring Cloud, Spring Data
  - Spring Batch, Spring Security
  - Spring for Android, Spring Web Flow
  - Spring Web Services, Spring LDAP

SET UP YOUR ENVIRONMENT

1. Java Application Server - **Tomcat**

2. Java Integrated Development Environment (IDE) - **Eclipse**

INSTALL ECLIPSE
MS WINDOWS

Connecting Eclipse and Tomcat

Benefits

- Start Tomcat from Eclipse
- Easily deploy applications directly to Tomcat

Downloading Spring JAR files

To Do List

1. Create Eclipse Project

2. Download Spring JAR Files

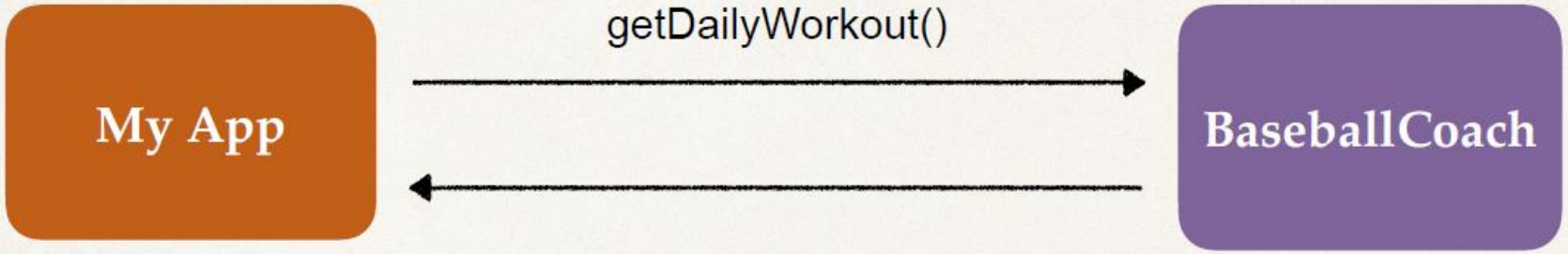
3. Add JAR files to Eclipse Project … *Build Path*

Inversion of Control

Inversion of Control (IoC)

The approach of outsourcing the construction and management of objects.

Coding Scenario

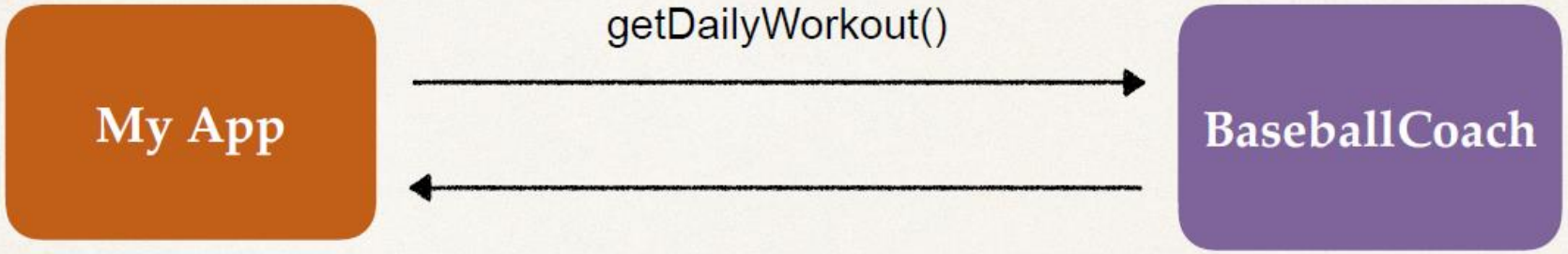


- App should be configurable
- Easily change the coach for another sport
  - Hockey, Cricket, Tennis, Gymnastics etc …

Code Demo

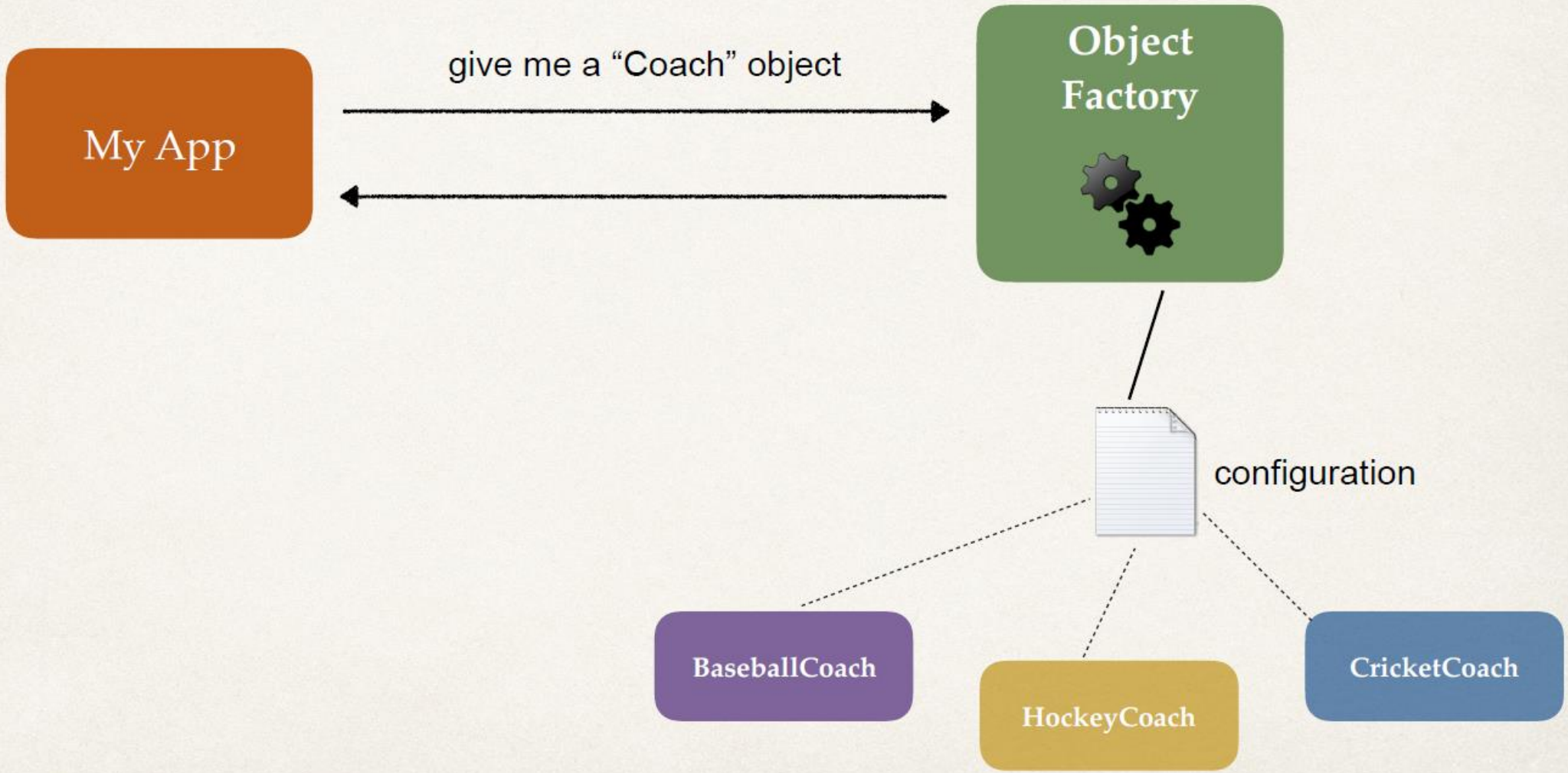
- **MyApp.java**: main method
- **BaseballCoach.java**
- **Coach.java**: interface after refactoring
- **TrackCoach.java**

Coding Scenario

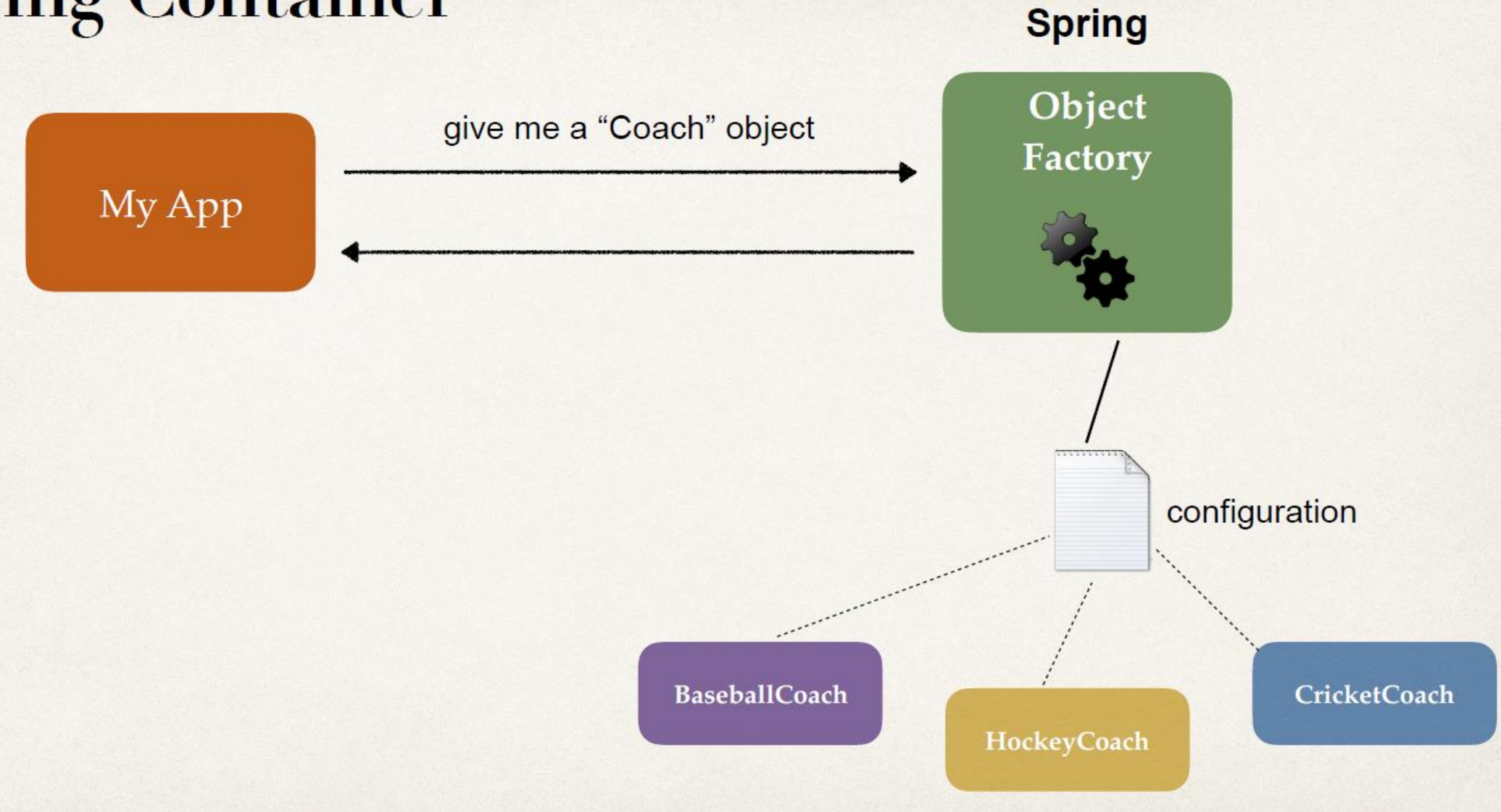


- App should be configurable
- Easily change the coach for another sport
  - Hockey, Cricket, Tennis, Gymnastics etc …

Ideal Solution



Spring Container



Spring Container

- Primary functions
  - Create and manage objects *(Inversion of Control)*
  - Inject object's dependencies *(Dependency Injection)*

Configuring Spring Container

- XML configuration file *(legacy, but most legacy apps still use this)*
- Java Annotations *(modern)*
- Java Source Code *(modern)*

Spring Development Process

1. Configure your Spring Beans
2. Create a Spring Container
3. Retrieve Beans from Spring Container

- **Step 1: Configure your Spring Beans**

```
<!-- Define your beans here -->
<bean id="myTrackCoach" class="com.springdemo.TrackCoach"></bean>
<bean id="myBaseBallCoach" class="com.springdemo.BaseballCoach"></bean>
```

Step 2: Create a Spring Container

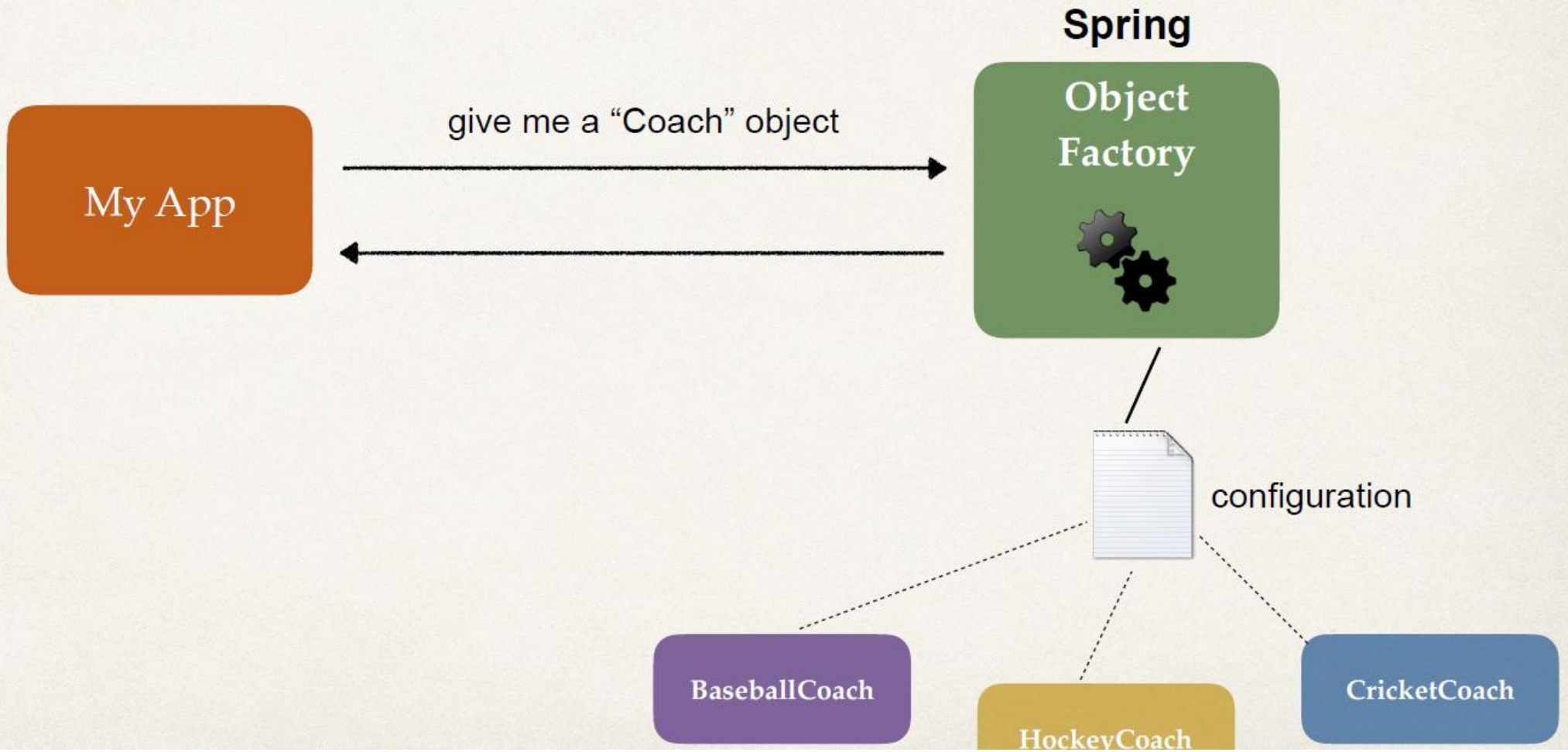
```
ClassPathXmlApplicationContext context =
    new ClassPathXmlApplicationContext("applicationContext.xml");
```

Step 2: Create a Spring Container

- Spring container is generically known as **ApplicationContext**

- Specialized implementations
  - ClassPathXmlApplicationContext
  - AnnotationConfigApplicationContext
  - GenericWebApplicationContext
  - others …

Step 3: Retrieve Beans from Container



- **Step 3: Retrieve Beans from Container**

```
//retrieve bean from spring container
Coach myTrackCoach=context.getBean("myTrackCoach",Coach.class);
//call methods on the bean
System.out.println(myTrackCoach.getDailyWorkout());


//retrieve bean from spring container
Coach myBaseBallCoach=context.getBean("myBaseBallCoach",Coach.class);
//call methods on the bean
System.out.println(myBaseBallCoach.getDailyWorkout());
```