Lập trình WWW *Java*

Spring IoC (Inversion of Control)
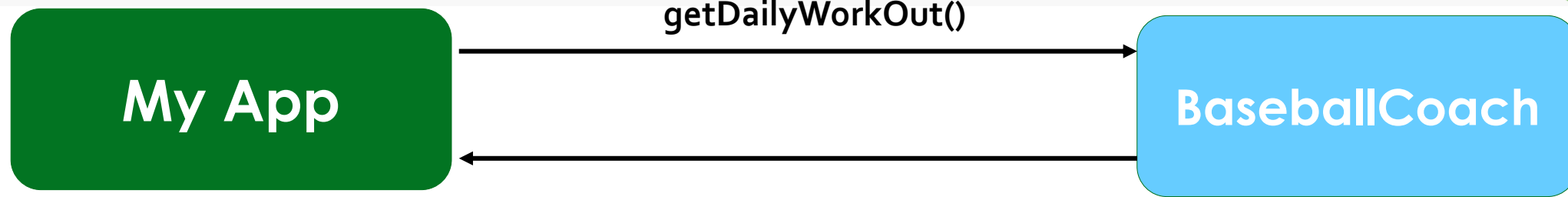
Inversion of Control (IoC)

Outsource to an object factory

The approach of outsourcing the construction and management of Objects

spring
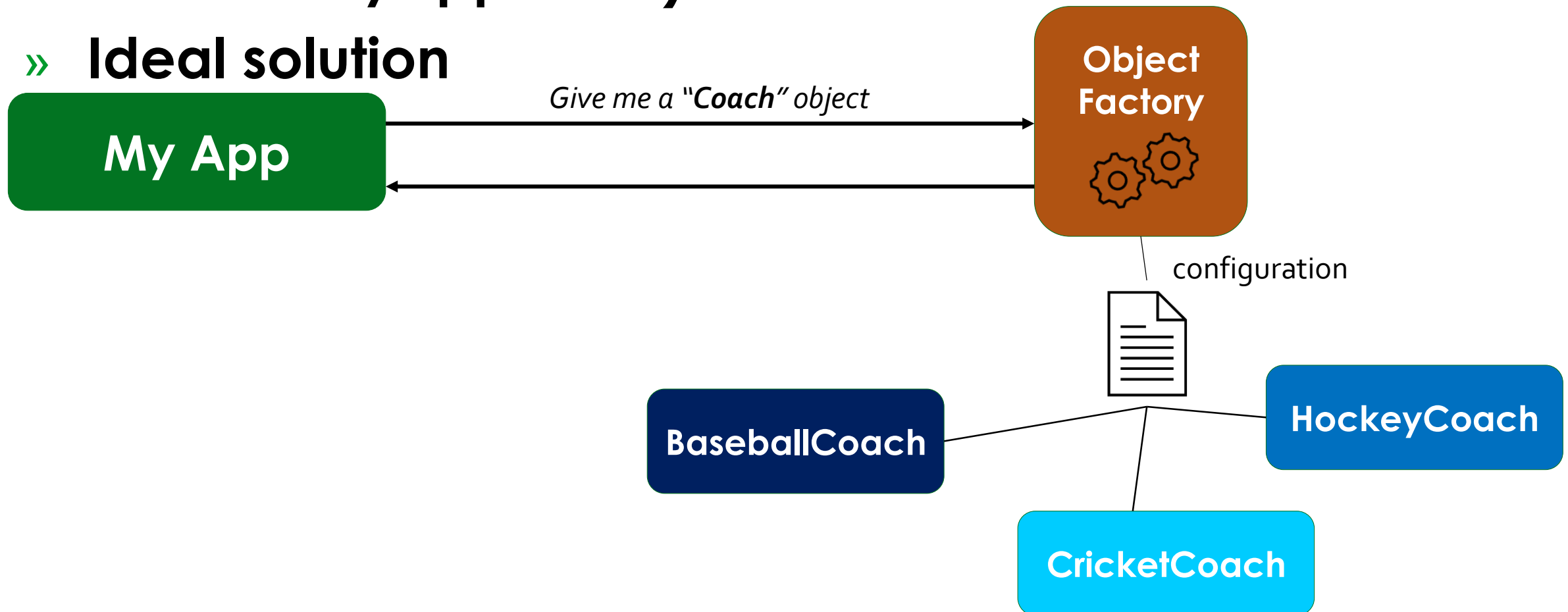
```
My App   ── getDailyWorkOut() ──>   BaseballCoach
         <──────────────────────
```
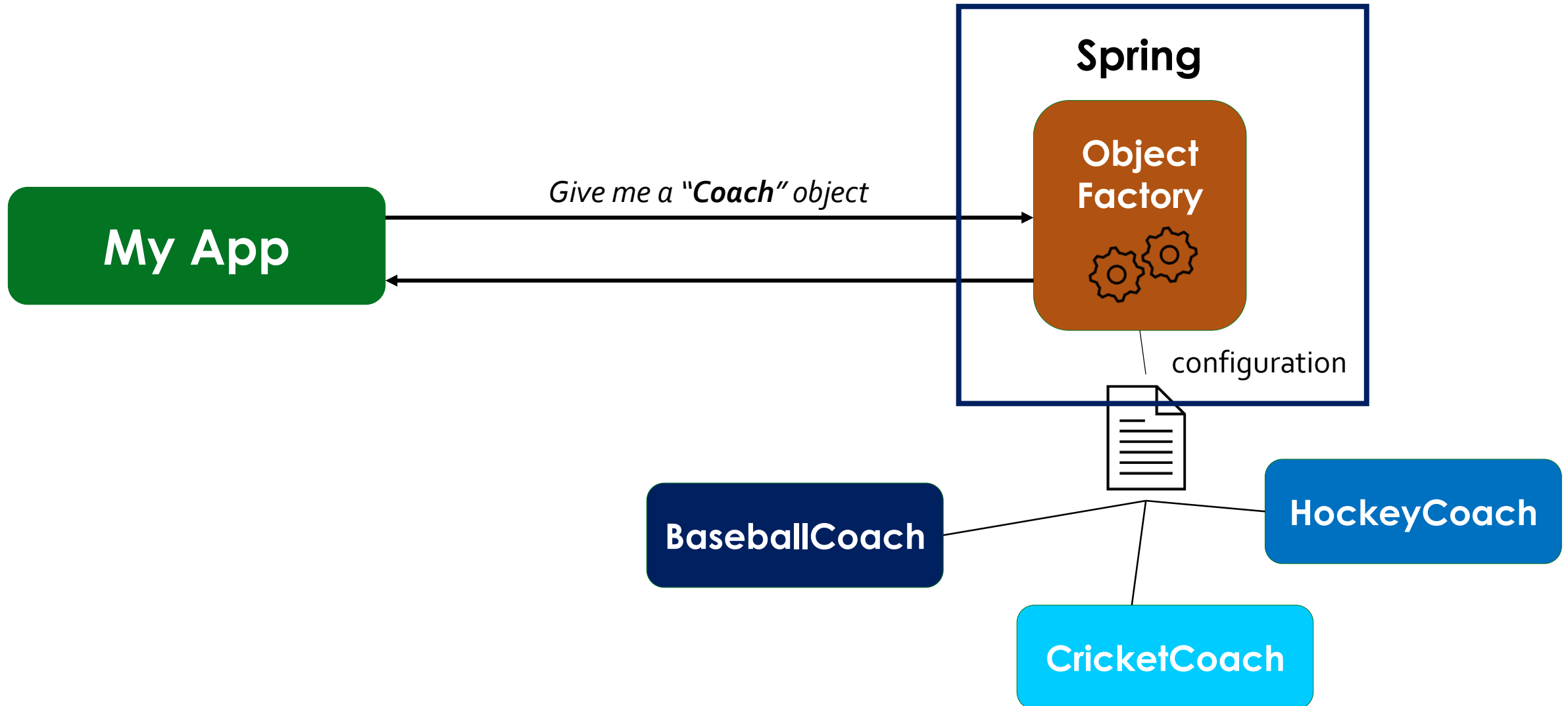
» **Easily change the coach** for other sport **Hockey, Cricket, Tennis etc.**

» The app should be **configurable**

» The app should be **configurable (I don't want to edit code in MyApp class)**

» **Ideal solution**

*Give me a "**Coach**" object*

My App

Object Factory

configuration

BaseballCoach

HockeyCoach

CricketCoach

IoC – Coding Scenario

Spring

Object Factory

*Give me a "**Coach**" object*

My App

configuration
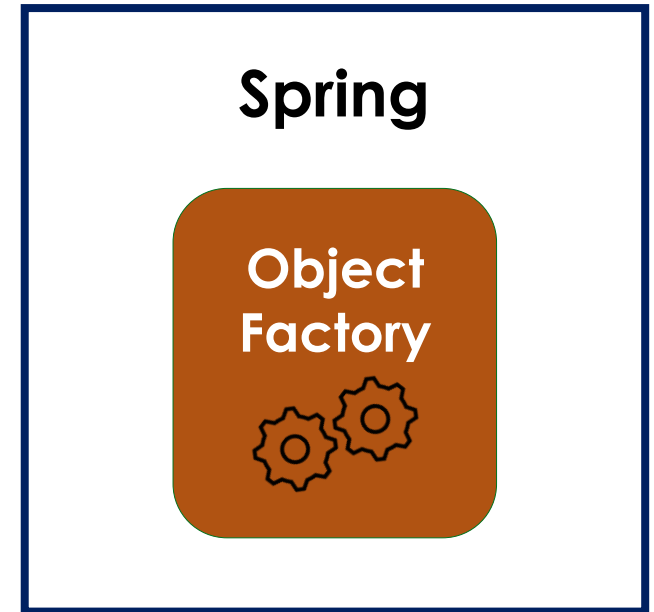
BaseballCoach

HockeyCoach

CricketCoach

6

Spring Container

spring

Primary functions:

» **Create** and **manage** objects (IoC)

» **Inject** object's **dependencies** (DI)

Spring

Object Factory

Configuring Spring Container
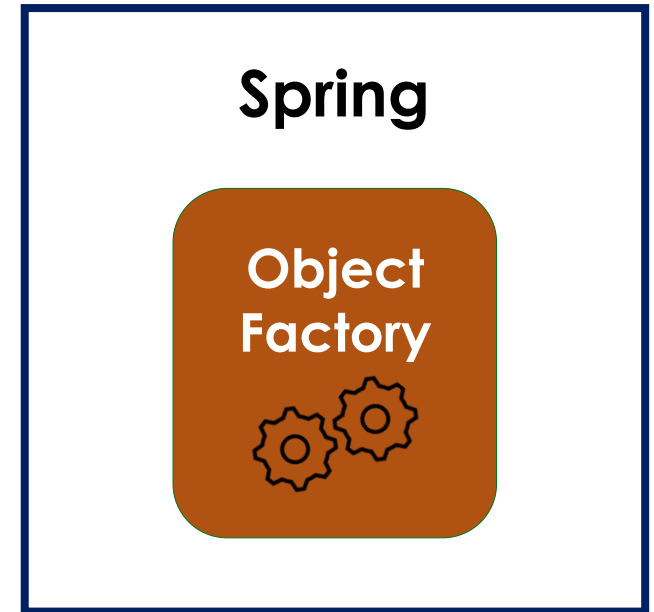
spring

» **XML configuration** file

» **Java Annotations**

» **Java Source Code**

Spring

Object Factory

Spring Development Process

» **Configuring** your Spring beans

» **Creating** a Spring container

» **Retrieving** the beans from the container

Spring

Object Factory

Step 1: Configuring your Spring beans

spring

File: applicationContext.xml:

```xml
<beans>

    <bean id="myCoach"
        class="com.se.springdemo.BaseballCoach">
    </bean>

</beans>
```

Step 2: Creating a Spring Container

Spring container is generally known as an **ApplicationContext**

Specialized implimentations:

- » ClassPathXmlApplicationContext ~ xml

- » AnnotationConfigApplicationContext ~ Annotaion

- » GenericWebApplicationContext ~ Java source code

Step 2: Creating a Spring Container

```java
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class SpringHelloApp {
    public static void main(String[] args) {
            // load the spring configuration file
            ClassPathXmlApplicationContext context=
                        new     ClassPathXmlApplicationContext
                            ("applicationContext.xml");
            //retrieve bean from spring container
            Coach theCoach= context.getBean("myCoach",Coach.class);
            //call method on the bean
            System.out.println(theCoach.getDailyWorkout());
            //System.out.println(theCoach.getFortune());
            //close the context
            context.close();
    }
```

```java
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class SpringHelloApp {
    public static void main(String[] args) {
            // load the spring configuration file
            ClassPathXmlApplicationContext context=
                        new   ClassPathXmlApplicationContext
                            ("applicationContext.xml")
            //retrieve bean from spring container
            Coach theCoach= context.getBean("myCoach",Coach.class);
            //call method on the bean
<beans>
    <bean id="myCoach"
        class="com.se.springdemo.BaseballCoach">
    </bean>
</beans>
```

Interface

Implement class

13

How to change another coach

Only change the implementation class (spring bean) in the config file

```xml
<bean id="myCoach"
      class="com.se.springdemo.BaseballCoach">
</bean>
```

Change to

```xml
<bean id="myCoach"
      class="com.se.springdemo.TrackCoach">
</bean>
```

QUESTIONS