**Method Injection**

**Method Injection**:

Method injection is different from Constructor Injection and Setter Injection. While in Constructor and Setter Injection, Spring creates the beans and injects them using the constructor or setter method, in Method Injection Spring overrides a given abstract method of an abstract class and provides an implementation of the overridden method.

This is an advanced form of dependency injection and should be used in very special cases as it involves byte-code manipulation by Spring.

A special case where Method Injection should be used is when a Prototype bean needs to use within a Singleton.

Suppose singleton bean A needs to use non-singleton (prototype) bean B, perhaps on each method invocation on A. The container creates the singleton bean A only once, and thus only gets one opportunity to set the properties. The container cannot provide bean A with a new instance of bean B every time one is needed.

We can actually inject our dependencies by calling any method on our class. So instead of using setter methods, we can actually use any method. All we have to do is simply give the @Autowired annotation on that method.

**Example**:

**File: Coach.java**:

**package** com.ruhul.odduu.methodinjection;

**public** **interface** Coach {

**public** String getDailyWorout();

**public** String getDailyFortune();

}

**File: FortuneService.java**:

**package** com.ruhul.odduu.methodinjection;

**public** **interface** FortuneService {

**public** String getFortune();

}

**File: HappyFortuneService.java**:

**package** com.ruhul.odduu.methodinjection;

**import** org.springframework.stereotype.Component;

@Component

**public** **class** HappyFortuneService **implements** FortuneService {

@Override

**public** String getFortune() {

**return** "Today is your lucky day!!!";

}

}

**File: TennisCoach.java**:

**package** com.ruhul.odduu.methodinjection;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Component;

@Component

**public** **class** TennisCoach **implements** Coach {

**private** FortuneService fortuneService;

// default constructor

**public** TennisCoach() {

System.***out***.println(">> TennisCoach: inside default constructor");

}

// method injection

@Autowired

**public** **void** doSomeCrazyStuff(FortuneService fortuneService) {

System.***out***.println(">> TennisCoach: inside doSomeCrazyStuff() method");

**this**.fortuneService = fortuneService;

}

@Override

**public** String getDailyWorout() {

**return** "Practice your backhand volley";

}

@Override

**public** String getDailyFortune() {

**return** fortuneService.getFortune();

}

}

**File: AnnotationDemoApp.java**:

**package** com.ruhul.odduu.methodinjection;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** AnnotationDemoApp {

**public** **static** **void** main(String[] args) {

// read spring congig file

ClassPathXmlApplicationContext context = **new** ClassPathXmlApplicationContext("applicationContextSetter.xml");

// get the bean from container

Coach theCoach = context.getBean("tennisCoach", Coach.**class**);

// call a method on the bean

System.***out***.println(theCoach.getDailyWorout());

// call method to get the daily fortune

System.***out***.println(theCoach.getDailyFortune());

// close the context

context.close();

}

}

**File: applicationContextMethod.java**:

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xmlns:context=*"http://www.springframework.org/schema/context"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context.xsd"*>

<!-- add entry to enable component scanning -->

<context:component-scan

base-package=*"com.ruhul.odduu.methodinjection"* />

</beans>

/\*

**Output**:

>> TennisCoach: inside default constructor

>> TennisCoach: inside doSomeCrazyStuff() method

Practice your backhand volley

Today is your lucky day!!!

\*/

Method Injection