

Spring Core – IoC



Presented by

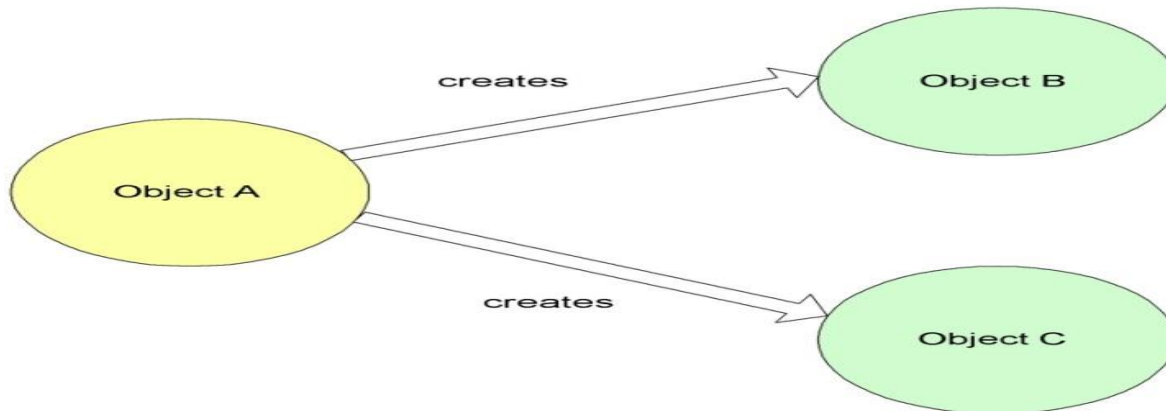


Dependency Injection & Inversion Of Control

Dependency : A POJO / bean / Entity

Non IOC – **Getting a POJO by pulling – Hard Coding**

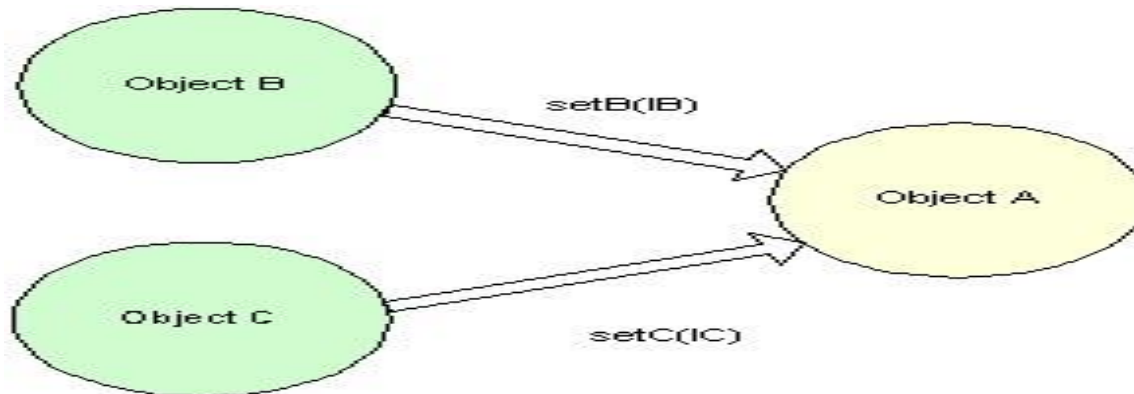
```
Employee emp1 = new Employee();  
Employee emp2 = new Employee();  
// Explicit creation
```



Inversion of Control

**IOC : Getting a POJO Object by Pushing on demand by
Spring Core Container**

```
Employee emp1 = (Employee)xxxxx.getBean("emp1");  
Employee emp2 = (Employee)xxxxx.getBean("emp2");
```



The act of Dependency Injection is known as 'wiring'.

Spring Core Container

Spring's Container uses IoC to manage components of the application.

- **Application context**
(org.springframework.context.ApplicationContext)
provides application framework services

Spring way of DI

- Java classes should be as independent as possible from each other
- Spring Framework injects these dependencies via the container.
- Piecing together all beans in the Spring Container is called **wiring**.
- Wiring can be done through xml.



Bean Configuration

Beans are listed in the configuration file

```
<?xml version= "1.0" encoding= "UTF-8"?>
<beans xmlns= "http://www.springframework.org/schema/beans"
  xmlns:xsi= "http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation= "http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

  <bean id= "helloWorld" class= "com.mycompany.springcore.hw.HelloWorld">
    <property name= "wish" value= "Hello World !"/>
  </bean>

</beans>
```

**HelloWorld application*

