International Journal on Recent and Innovation Trends in Computing and Communication ISSN 2321 – 8169
Volume: 1 Issue: 4 339 – 342

SPRING TECHNOLOGY AND USE IN BUSINESS APPLICATION

Ashish pandey
M.Tech (CSE)
Bhagwant University
Ajmer (Raj), India
Email: aspokn@gmail.com

Abstract— **this paper discusses spring technology and use in business application development. Focus on object creation, dependency satisfaction & life-cycle management must be delegated by spring framework (IOC, AOP, TX, and ORM). Spring doesn't confine itself to any specific domain. It can be use in a simple console application or in a distributed business application. Development point of view Spring reduces 70% of coding efforts [1]. It helps in object creation, transaction- management, connectivity with database Testing and deploying an application.**

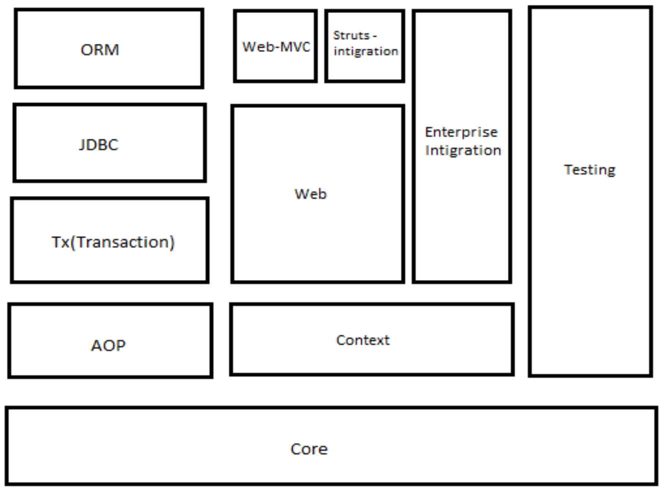
Index Terms— **spring, IOC, AOP, TX**

*****

INTRODUCTION

Spring is general purpose framework which is based on two concepts name **IOC** (Inversion of control) and **AOP** (Aspect oriented programming). [1]
Spring doesn't confine itself to any specific domain. It can be use in a simple console application, in a web-application or in a distributed-application.

This framework is divided in different modules which can be use individually or collectively [1].



Modules of spring

Core: This module provides the implementation of basic IOC container

Context: This module is build over core and provides the implementation of an advanced IOC container

AOP: Aspect oriented programming module provides the implementation of spring.

Core, AOP & Context: These three modules represents the basic concepts of spring framework rest of modules represent the application of these concepts into different domain

TX (Transaction): This module provides the implementation of transaction management using the concept of AOP.

JDBC: This module provides the implementation for JDBC operations.

ORM (Object Relational Mapping): This module facilitates integration of spring framework to ORM framework. Such as hibernate, ibatis etc.

WEB: This module provides the implementation of common operations of web-application such as transfer of request data to module objects, type conversion, exception handling, file uploading and downloading.

WEB-MVC: This module provides implementation of MVC for developing web-application

Struts-integration: This module facilitates integration of spring framework to struts framework.

Enterprise-Integration: This module facilitates integration to enterprise services such as JMS, JNDI, Web-services, java-mail etc.

Testing: This module facilitate IOC base unit testing & integration Testing.

CORE : IOC (INVERSION OF CONTROL)

One of the basic concepts of spring framework is IOC this concept deals with the creation & use of objects[2].
This concept states the application-program must only be concerns with the use of object creation, dependency satisfaction & life-cycle management. Must be delegated to some other party. This is called Ioc-Container in spring.

Implementation of IOC is provided through:

(i) ***DI (dependency Injection) :*** dependency Injection is satisfy dependency un-asked (with-out asking)

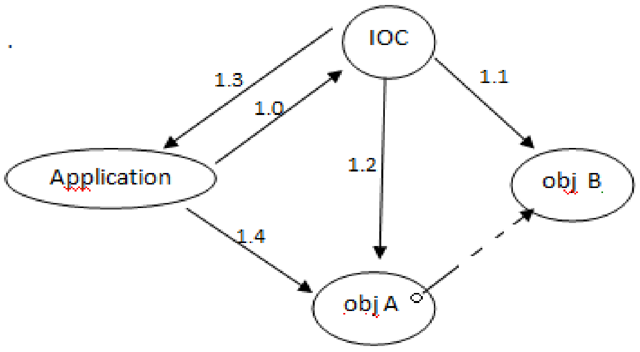
(ii) ***DL (Dependence Look-up) :*** dependency Look-up are satisfy only when the container is asked

DI & DL are mean and IOC is the end.
Spring IOC implementation support only DI (Dependency Injection)

IOC Container use two-approaches of injection (injecting) dependency
(i) Setter Injection
(ii) Construction Injection

Dependency of object can be of two types
(i) *Mandatory:* Mandatory dependencies are injected by **constructor.**

Example: *Let object 'A' has a mandatory dependency on object 'B'. This dependency will be injected by IOC container. Using constructor injection as follow*



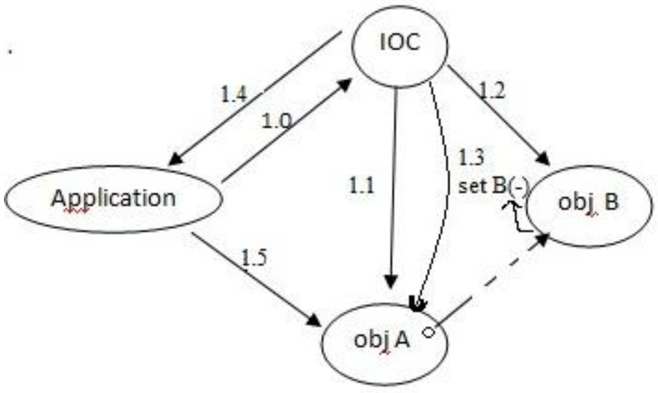
Dependency satisfy by constructor injection

1.0 object 'A' is requested
1.1 object 'B' is created to satisfy the dependency of 'A'
1.2 objects 'A' is created & reference of 'B' is provided as constructor parameter
1.3 reference of 'A' is returned
1.4 'A' object is used

Note: In this case object 'A' will be created if only if object 'B' is created.

(ii) *Optional:* Optional dependencies are injected through **setter** injection.

Example: *Let object 'A' has optional dependency on object 'B' these dependencies will be injected by IOC container using setter injection as follow*

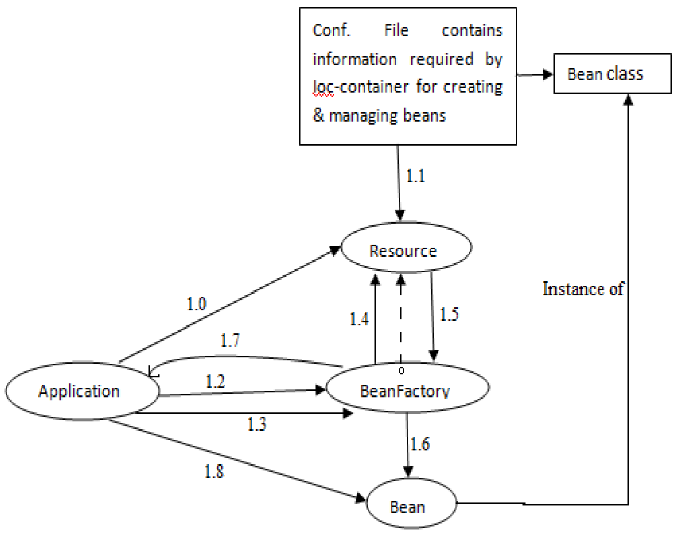


Dependency satisfy by Setter injection

1.0 object 'A' is requested
1.1 object 'A' is created
1.2 'B' object is created to satisfy the dependency of object 'A'
1.3 setB (---), reference of 'B' is made available to object 'A'
1.4 reference of 'A' is returned
1.5 'A' object is used.

Note: In this case object 'A' will be made available even object 'B' is not created.

ARCHITECTURE OF SPRING


Conf. File contains information required by Ioc-container for creating & managing beans
Bean class
1.1
Resource
Instance of
1.0
1.4
1.5
1.7
1.2
Application
BeanFactory
1.3
1.6
1.8
Bean

Basic Architecture of Spring

1.0 Resource object is created

1.1 Read configuration from configuration file.

1.2 BeanFactory is created & reference object provided to it

1.3 A bean requested from the container

1.4 Configuration of bean is requested from the resource object

1.5 Configuration information is provided to the BeanFactory

1.6Bean is instantiated & its dependency are satisfied

1.7bean reference is returned.

1.8 Bean is used

Terminology of spring:

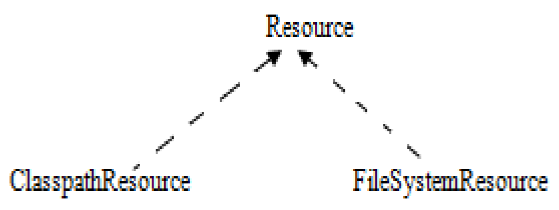
Bean: In spring a bean is an object which is created and managed by Ioc-container.

BeanFactory: BeanFactory is an interface which describes functionality of basic Ioc-container

The most commonly used method of interface is getBean (). This method is used to requested a Bean from Ioc-container

Configuration Information: Information required by Ioc-container for creating & managing beans is called configuration information. Configuration information is provided through annotation and through xml file.

Resource: Resource is an interface which provides method for managing configuration-information in object form. Multiple implementation of this interface are provided by the framework



Classpathresource: Classpathresource is used when configuration file can be located using class path or inside source folder

FileSystemResource: FileSystemResource is used when configuration file can be located outside source folder using path

SPRING CONFIGURATION

Spring is managed by configuration file that contain information required by Ioc-container for creating & managing Beans

Spring configuration [1] in programmatic means of Business application

MyShape_Draw
- ➔ Org.pandey
  - ➔ Triangle.jawa
  - ➔ DrawingApp.java
  - ➔ MyFactory.java
- ➔ JRE System Library
- ➔ Spring_Lib
  - ➔ MySpring.xml

Triangle.java

```
package org.pandey;

Public class Triangle {

public void MyDrawing(){
System.out.println("Triangle is Drawing started soon...");
}

}
```

DrawingApp.java

```
package org.pandey;

import org.springframework.beans.factory.BeanFactory;[1]
public class DrawingApp {

        public static void main(String[] ss) {
                BeanFactory f =
Myfactory.getBeanFactory();
                Triangle tri = (Triangle) f.getBean("draw");
                tri.MyDrawing();
        }

}
```

MyFactory.java

```
package org.pandey;
```

```
import org.springframework.beans.factory.BeanFactory;[1]
import org.springframework.beans.factory.xml.*;
import org.springframework.core.io.FileSystemResource;[1]

public class Myfactory {
static BeanFactory factory;
static {
factory = new XmlBeanFactory(new
FileSystemResource("myspring.xml"));
}

public static BeanFactory getBeanFactory() {
return factory;
}
}
```

MySpring.xml

```
<? 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.o
rg/schema/beans
http://www.springframework.org/schema/beans/spring-
beans.xsd">[1]

<bean id="draw" class="org.pandey.Triangle"/>

</beans>
```

V Advantages and Disadvantages of Spring

Advantages of spring

- Spring is Lightweight container
- Spring in not depended on application server like : EJB, JNDI etc
- Objects are created lazily, singleton – by configuration or annotation [5].
- Component can added in declarative manner [5].
- Application code is much easier in testing : unit testing
- With Dependency Injection approach, dependencies are explicit, and satisfying in constructor or Setter properties
- Not required special deployment steps [5]
- Productivity of programmer is increase as they are only concern with the use of objects.
- Reusability of object is may possible because object creation and managed by single party in the application
- Maintenance of application is simplify because dependency are managed declaratively

Disadvantages of Hibernate

- ***Need to learn lots of API***: A lot of effort is required to learn spring. So, not very easy to learn spring for a beginner.
- ***Debugging:*** sometime debugging and performance tuning take time.

VI Conclusion

This paper has illustrated an introduction of what spring can do. What we can achieve using spring in business application. And how using spring deliver high performance. We explain architecture of spring and describe CORE, AOP, transaction, JDBC, ORM, WEB, WEB-MVC, Struts-integration, Enterprise-integration, and Testing, Developer point of view Spring reduce 70% of coding effort. It helps in coding (creation of objects), reusability, maintenance and unit- testing of an application [1]. Spring also facilitate to integration with hibernate, ibatis and other frameworks.

VII References:

[1] *www.**springs**ource.org*
[2] Spring In Action 3.0: Craig Walls.
[3] www.mkyong.com
[4] javabrains.koushik.org
[5] www.techfaq360.com
[6] Pro Springs 3.0: Apress

VIII Author:

Ashish pandey is pursuing **M.Tech** in Computer Science from Bhagwant University, Ajmer (Rajasthan), India. He is very vast experience in computer science and engineering area and participated in academic and industry related real time projects. He is working as freelancer and engages with enterprise application development industry. He can be reached at: aspokn@gmail.com