Intro:

What the spring bean factory is and some fundamental concepts about Spring.

When we talk about Spring first thing comes to our mind is DI(Dependency Injection) and all the DI is possible because Spring container which contains Spring beans.

What does mean by container take an example of servlet container, tomcat is servlet container what does that mean ??

Tomacat create servlet objects which are required to run an application we mainely configure our servlets using XML file which servlet reads and instantiate servlet objects.

Spring does the same like tomcat but spring is not the container for servlet, it’s a container for bean. It creates beans which are simple POJO and contains them.

Object

Object

Object

Object

Object

Object

Spring Container

Object

Spring Container takes care of instantiation of objects,destruction of objects etc.

We can also have some objects outside spring container

Spring manages the lifecycle of an object.the way it works is first container creates the object.

Lets say we don’t have any spring container then we have to create object using

A a = new A();

But in case of spring container we don have to create the object ourself, we ask spring container to create the object for us.

So this is called as factory pattern.

Lets discuss what that mean.

We have a Factory class which used to produce the objects and send it across to us.

But how its possible and does factory comes to know that what properties values should it have to instantiate the object.

This all is done by XML configuration file or Annotation configuration.

Factory Class

XML Configuration file

Our Class

New Object

Here our class need to have one object, the factory class read the xml file and instantiate the object for us and transfer the same to us.

Now the same happened in Spring using BeanFactory.

Our Class

Spring Bean Factory

Spring Configuration file

Spring Bean

Let us take a look at a working Eclipse IDE in place and take the following steps to create a Spring application −

|  |  |
| --- | --- |
| Steps | Description |
| 1 | Create a project with a name *SpringExample* and create a packagecom.tutorialspoint under the **src** folder in the created project. |
| 2 | Add the required Spring libraries using Add External JARs option as explained in the Spring Hello World Example chapter. |
| 3 | Create Java classes *HelloWorld* and *MainApp* under the *com.piyushpackage*. |
| 4 | Create Beans configuration file *Beans.xml* under the **src** folder. |
| 5 | The final step is to create the content of all the Java files and Bean Configuration file. Finally, run the application as explained below. |

Here is the content of **HelloWorld.java** file −

package com.piyush;

public class HelloWorld {

private String message;

public void setMessage(String message){

this.message = message;

}

public void getMessage(){

System.out.println("Your Message : " + message);

}

}

Following is the content of the second file **MainApp.java**

package com.piyush;

import org.springframework.beans.factory.InitializingBean;

import org.springframework.beans.factory.xml.XmlBeanFactory;

import org.springframework.core.io.ClassPathResource;

public class MainApp {

public static void main(String[] args) {

XmlBeanFactory factory = new XmlBeanFactory (new ClassPathResource("Beans.xml"));

HelloWorld obj = (HelloWorld) factory.getBean("helloWorld");

obj.getMessage();

}

}

Following two important points should be noted about the main program −

* The first step is to create a factory object where we used the framework APIXmlBeanFactory() to create the factory bean andClassPathResource() API to load the bean configuration file available in CLASSPATH. The XmlBeanFactory() API takes care of creating and initializing all the objects, i.e. beans mentioned in the configuration file.
* The second step is used to get the required bean using getBean() method of the created bean factory object. This method uses bean ID to return a generic object, which finally can be casted to the actual object. Once you have the object, you can use this object to call any class method.

Following is the content of the bean configuration file **Beans.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.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

<bean id = "helloWorld" class = "com.piyush.HelloWorld">

<property name = "message" value = "Hello World!"/>

</bean>

</beans>

Once you are done with creating the source and the bean configuration files, let us run the application. If everything is fine with your application, it will print the following message −

Your Message : Hello World!