DEPENDENCY CHECK

SpringTutors January 10, 2016 Features 69 Views

Dependency Check

We can perform injection in two ways either via setter or constructor. When we apply constructor injection it is mandatory for us to configure the constructor in spring bean configuration file as <constructor-arg>. But in case of setter injection it is optional to configure the setter as <property name>

By default dependency check is disabled in spring for setter. To make setter injection mandatory we have to enable the dependency check by using an attribute called dependency-check at bean tag level with different modes.

Three different Modes are –

1. simple
2. object
3. all

Lets see an example

```
class Person
{
private int id;
private Address address;
public void setId(int id)
{
this.id=id;
}
public void setAddress(Address address)
{
this.address=address;
}
}
```

```
class Address
{
private String cityName;
public void setCityName(String cityName)
{
this.cityName=cityName;
}
}
```

Now configure the above class as bean

```
<bean id="person" class="Person"/>
<bean id="address" class="Address"/>
```

Now if we configure the bean like above for the class Person and class Address IOC container will create the object for 'Person' and 'Address' because we have applied setter injection which is optional. By calling the default constructor of 'Person' and 'Address' Spring will create the object. To make the setter injection mandatory we have to on the dependency check which is by default off like following-

Now when dependency mode is "simple"-

```
<bean id="person" class="Person" dependency-check="simple">
<property name="id" value="554"/>
</bean>
<bean id="address" class="Address"/>
```

In this case IOC container will see that dependency is of simple type then it goes to respective bean class and find the attribute name of primitive type for which setter is defined and comes to the configuration file and search the property name in bean configuration if it is found then inject the value to the respective bean otherwise throws an exception.

Note:- In this case IOC will only checks the simple type attributes name for which setter is defined. It does not consider the object type for which setter is defined or not.

Now when the dependency mode is "object"-

```
<bean id="person" class="Person" dependency-check="object">
<property name="id" value="554"/>
</bean>
<bean id="address" class="Address"/>
```

In this case IOC container goes to the respective bean class and finds the name of the object type for which setter is defined and comes to the configuration file and checks the corresponding name in property of the bean if found then inject the reference otherwise throws an exception.

Note:- In this case IOC will only checks the object type attributes name for which setter is defined. It does not consider the simple type for which setter is defined or not.

Now when the dependency mode is "all"-

```
<bean id="person" class="Person" dependency-check="all">
<property name="id" value="554"/>
<property name="address" ref="address"/>
</bean>
<bean id="address" class="Address"/>
```

In this case IOC container will goes to the respective bean class and looks for both primitive type attribute name and object type attribute name for which setter is defined and then comes and checks in configuration file property name in respective bean if found both then inject otherwise throws an exception.

Note:- In this case IOC will only checks the object type attributes name for which setter is defined and the simple type attribute name for which setter is defined. It means for both IOC will check the dependency .

For the above classes bean configuration is-

```
<bean id="person" class="Person" dependency-check="all">
<property name="id" value="554"/>
<property name="address" ref="address"/>
<bean id="address" class="Address">
<property name="cityName" value="hyderabad"/>
</bean>
```