**Dependency Injection:**

Consider the following code:

package com.dev.springExample;

class Employee

{

private int id;

private String name;

public Employee()

{

System.out.println("def cons");

}

public Employee(int id)

{

this.id = id;

}

public Employee(String name)

{

this.name = name;

}

public Employee(int id, String name)

{

this.id = id;

this.name = name;

}

void show()

{

System.out.println(id+" "+name);

}

}

Now, the xml file which which invoke the constructor **(So far, dependency between two classes are not there)**

**<?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:p="http://www.springframework.org/schema/p"**

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

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

**<bean id="e" class="com.javatpoint.Employee">**

**<constructor-arg value="10" type="int"></constructor-arg>**

**</bean>**

**Now, what does it do?**

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

It makes the beans schema the default for this xml file, which will allow referring to the elements in this beans schema without a namespace prefix.