/**

* Author: Vikash Ranjan

*/

========== ======== **Spring IoC container** ==========

The IoC (Inversion of control) container is responsible to instantiate, configure, assemble the objects and manage their complete life cycle from creation till destruction. The spring container uses DI to manage the components that make up an application. These objects are called Spring Beans.

The IoC container gets informations from the XML Configuration file and works accordingly. The main tasks performed by IoC container are:

-> to instantiate the application class

-> to configure the object

- > to assemble the dependencies between the objects

There are two types of IoC containers. They are:

1) **BeanFactory : org.springframework.beans.factory.BeanFactory**

2) **ApplicationContext : org.springframework.context.ApplicationContext**

The ApplicationContext interface is built on top of the BeanFactory interface. It adds some extra functionality than BeanFactory such as simple integration with Spring's AOP, message resource handling (for I18N), event propagation, application layer specific context for web application. So it is better to use ApplicationContext than BeanFactory.

Using BeanFactory :

The XmlBeanFactory is the implementation class for the BeanFactory interface. To use the BeanFactory, we need to create the instance of XmlBeanFactory class as given below.

```
Resource resource=new ClassPathResource("applicationContext.xml");

BeanFactory factory=new XmlBeanFactory(resource);
```

Using ApplicationContex:

The ClassPathXmlApplicationContext class is one of the most common implementation classes of ApplicationContext interface. We need to instantiate the ClassPathXmlApplicationContext class to use the ApplicationContext as given below.

```
ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
```

Some of the useful ApplicationContext implementations that we use are;

1) **AnnotationConfigApplicationContext** : If we are using Spring in standalone java applications and using annotations for Configuration, then we can use this to initialize the container and get the bean objects.

2) ClassPathXmlApplicationContext : If we have spring bean configuration xml file in standalone application, then we can use this class to load the file and get the container object.

3**) FileSystemXmlApplicationContext** : This is similar to ClassPathXmlApplicationContext except that the xml configuration file can be loaded from anywhere in the file system.

4) **XmlWebApplicationContext** : It is used for web applications.