spring®

Spring Framework

What is springs in java ?

- Spring is **a powerful lightweight application development framework used for Java Enterprise Edition (JEE)**.
- Spring is an open-source framework for building enterprise java applications.
- We can say the spring is the framework of frameworks
- Spring has several modules which helps us to build java applications in a simple way and removes many complexities..

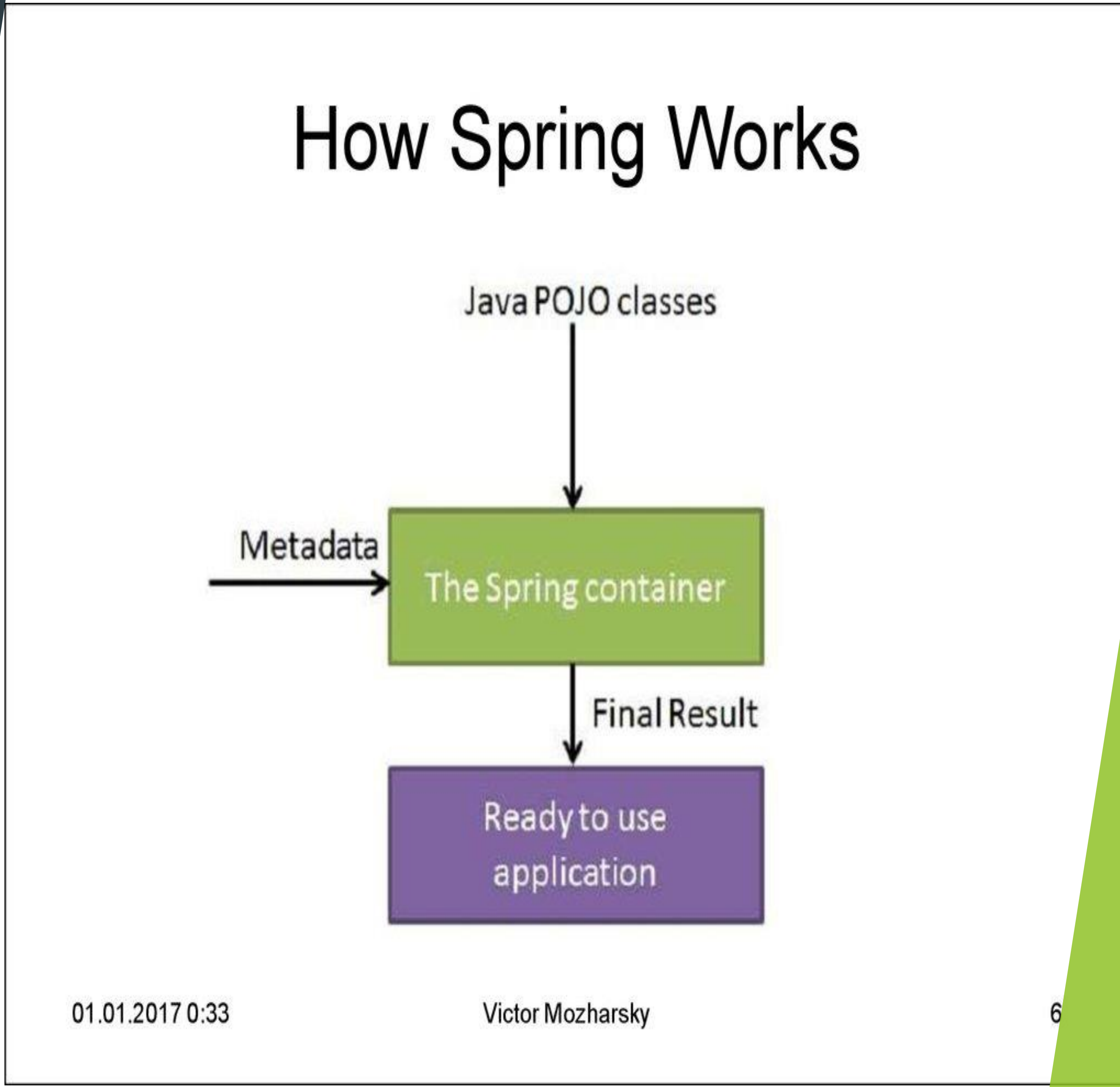
Why we have to go for springs

- spring is a light-weight framework that can be used to create scalable , secure and robust enterprise web applications.
- Spring aims to simplify the complexity of java application development process…
- the spring framework is also the base that powers all the other spring based projects such as:
- 1. spring Boot
- 2. Spring cloud
- 3. spring GraphQl et

Spring – Inversion of Control(IoC)

- Spring IoC Container is the core of Spring Framework…
- IoC containers creates objects for us, so that we need not to create manually..
- IoC(Inversion of control) Container is responsible to instantiate , configure and assemble the objects..
- To create objects for us, IoC container needs some information.. We provide that configuration in an xml file…..
- The main task performed by the Ioc Container are:
- a. To instantiate the application class
- b. To configure the object
- c. To assemble the dependencies between the objects...

Architecture of Spring IoC Container



How Spring Works

Java POJO classes

Metadata

The Spring container

Final Result

Ready to use application

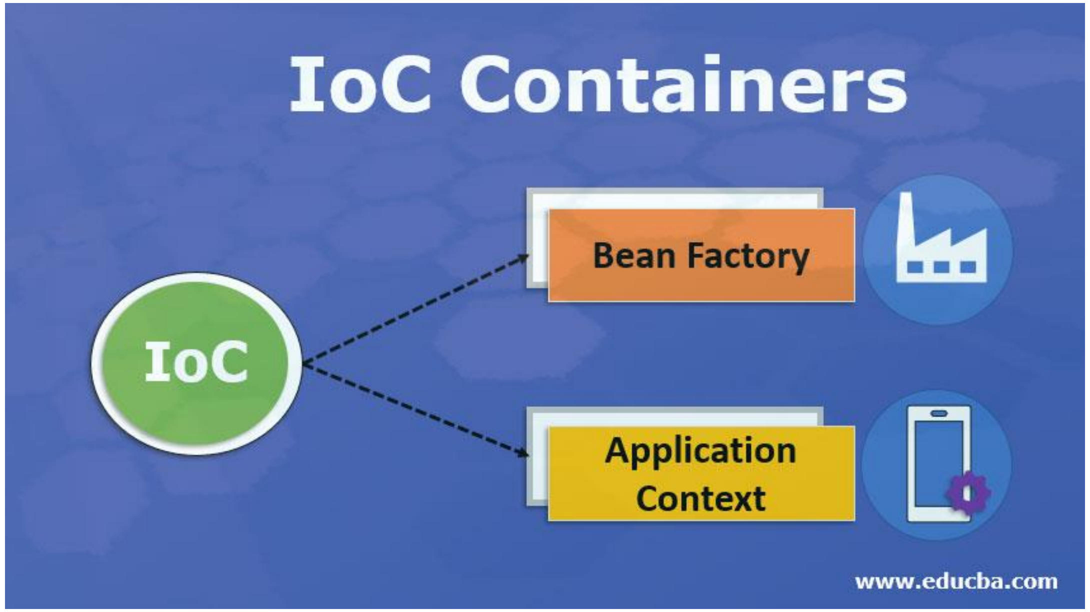
01.01.2017 0:33

Victor Mozharsky

6

Types of IoC Containers

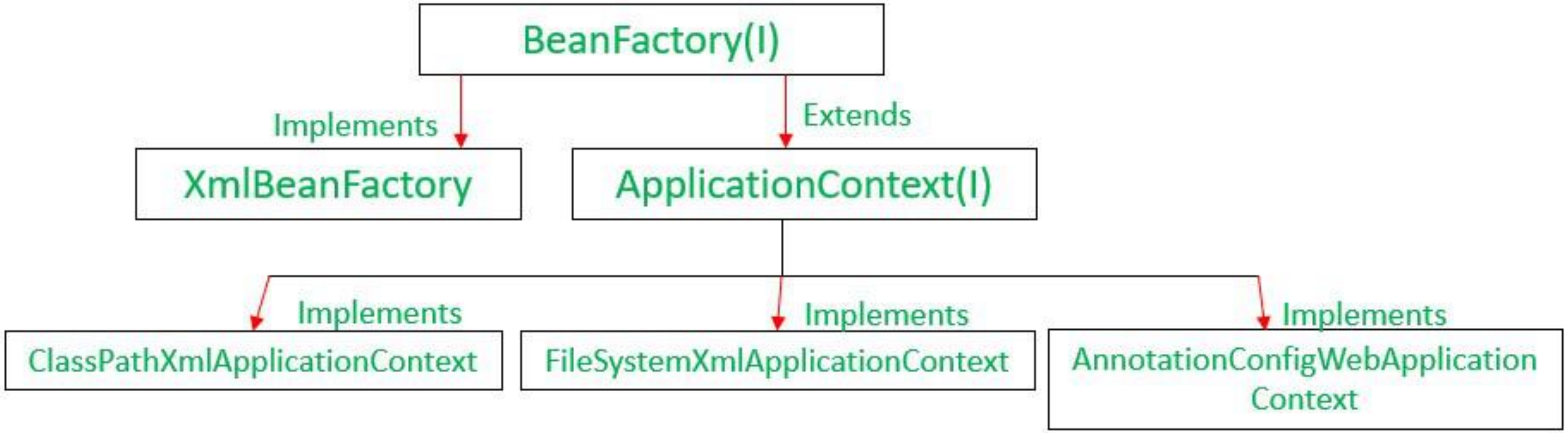
- There are two types of IoC containers
- 1. Bean Factory
- 2. Application Context



Dependency Injection

- When the dependency is injected by the Ioc container then it is called as dependency injection

Architecture of Bean factory and Application context



BeanFactory(I)

Implements

Extends

XmlBeanFactory

ApplicationContext(I)

Implements

Implements

Implements

ClassPathXmlApplicationContext

FileSystemXmlApplicationContext

AnnotationConfigWebApplication Context

Bean Factory

- The BeanFactory is **the actual container which instantiates, configures, and manages a number of beans**. These beans typically collaborate with one another, and thus have dependencies between themselves.

Application-Context

- The Application Context is **Spring's advanced container**. Similar to BeanFactory, it can load bean definitions, wire beans together, and dispense beans upon request.
- The container gets its instructions on what objects to instantiate, configure, and assemble by reading configuration metadata.

Difference between bean factory and application context

- Bean Factory :- Bean factory is an interface. Bean Factory supports lazy Loading. It doesn’t support annotation based dependency injection .

- Application Context :- Application Context is an interface which implements bean factory. Application Context supports aggressive loading. It supports annotation based dependency injection….

Different bean scopes in Spring

- Singleton :- The bean instance will be only one and same instance will be returned by the IoC container. It is the default scope
- Prototype :- The bean instance will be created each time newly whenever we request.
- Request:- The bean instance will be created as per HTTP request
- Session:- The bean instance will be created as per HTTP session.
- GlobalSession:- The bean instance will be created as per HTTP global session. It can be used in portlet context onlhy

Different ways of dependency injection using XML file Configuration

- **1.Constructor based dependency injection** :- we can inject the dependency by constructor by using <constructor-arg> (sub element of <bean>) is used for constructor injection…
- Here we can inject primitive and String based values, dependent object collection values etc…
- For Constructor injection we have to declare a suitable constructor to inject the targeted values..
- After the creation of class we have to create gag of constructor in xml file inside specific bean and we can insert values by using attribute and object by using (ref) attribute..
- Example:-

```
<constructor-arg index="0" value = "a"> </constructor-arg>

<constructor-arg index="1" value = "10"> </constructor-arg>
```

- **2. By using setter** :- we can inject the dependency by using setter method also..
- The <property> statement of <bean>is used for setter injection
- By using setter we can inject primitive and string-based values, dependent object , collection values etc.
- For setter injection , we need to declare setter method for each property of specific class
- Then inside the respectice bean tag we have to declare <property> sub element. Where we can pass the value to the properties by providing its variable name to name attribute.
- Here “name” points the attribute where value specifies the data which is going to assign to it

Different ways of dependency injection using Annotation Configuration

- We can do dependency injection with the help of annotations…..
- There are many annotations which helps us to perform dependency injection. Some of them are below:-
- @Component:- we use this annotation on the above of the class to get the object of that class…
- @autowired:- we use this annotation on the above of dependent object, so that we can get that object
- @Configuration:- we use this annotation on the above of the helper class to configure
- @component-scan:-we use this annotation to specify the package, so that we can get object of the classes of that package