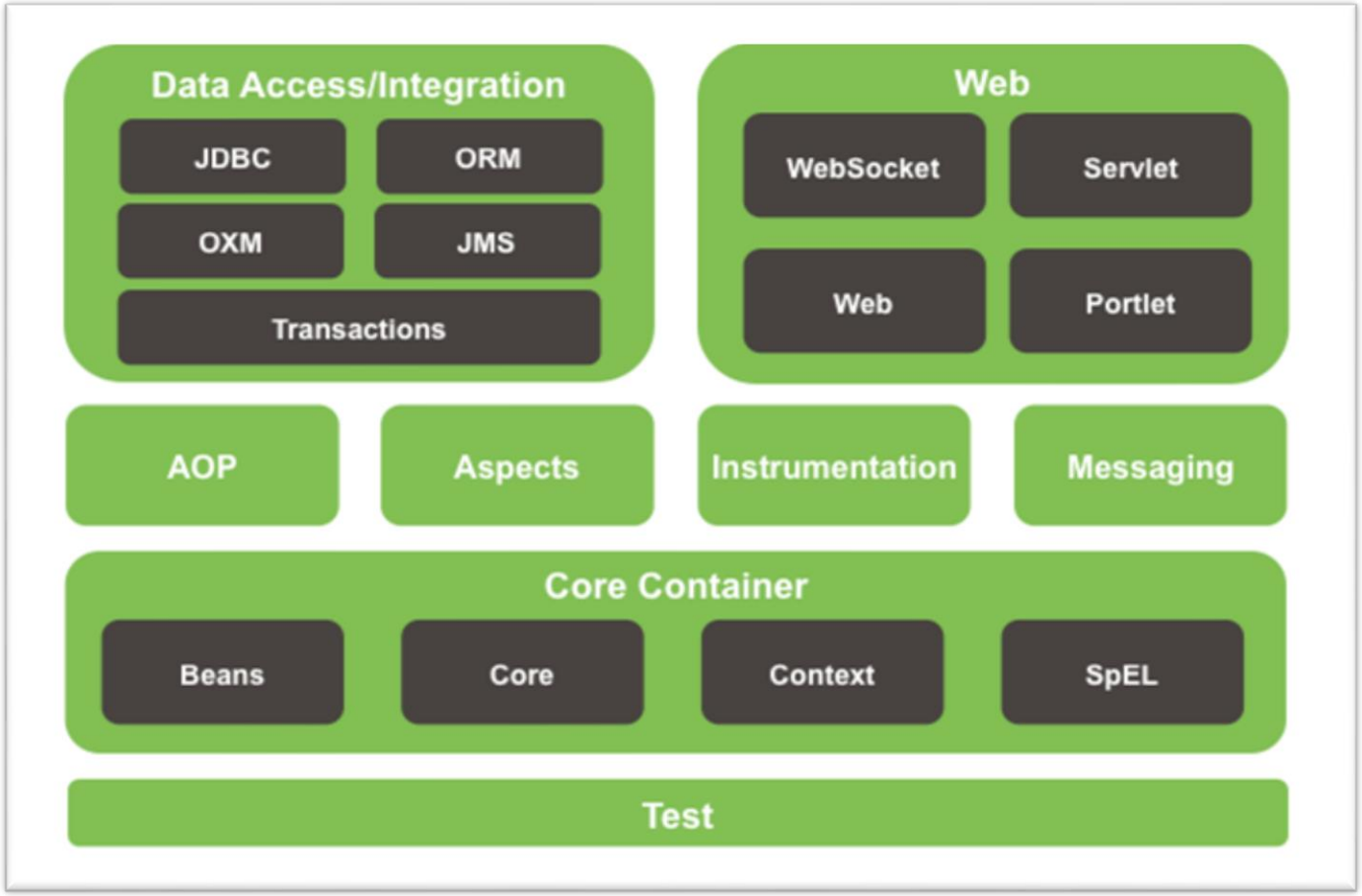
Welcome To

Spring Boot & Microservices

Introduction

- Java Based Open Source Framework
- Enterprise Application Development
- Modularized Framework
- Simplicity of POJOS
- Dependency Injection & Inversion of Control (IOC)
- Lightweight Framework

Spring Architecture



- Spring Core
- Spring Web
- Spring DAO
- Spring AOP

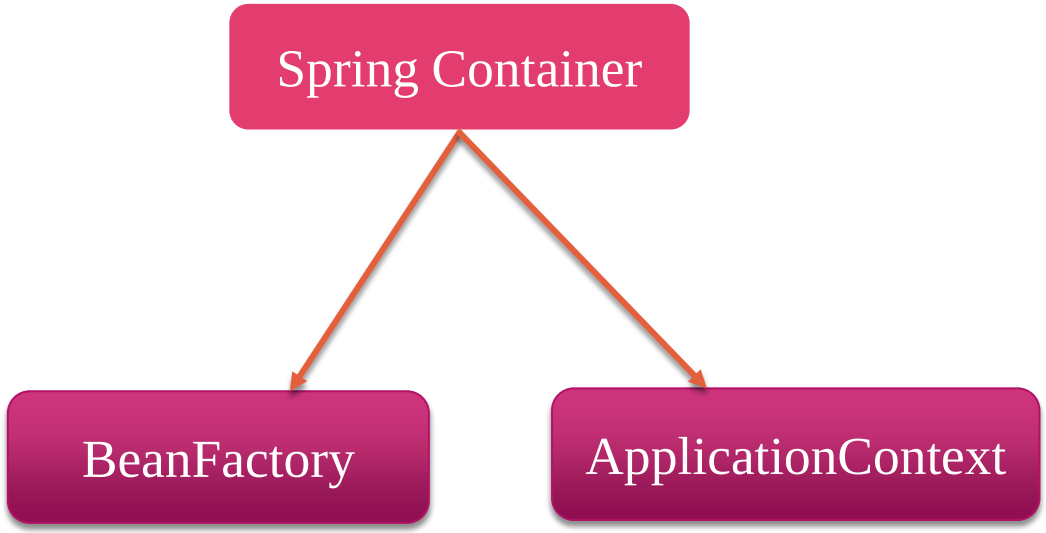
Spring Core

- Spring Core Module is base Module for remaining Modules of Spring Framework.
- Spring Core is the central part of spring framework to work with spring container. It manages how the beans are created, configured in a spring application.
- This module makes spring application as ”**Light -Weight**” by providing loose coupling between the objects.
- This module provides services like **Dependency Injection(DI),** Email, I18N, AOP programming etc.

Spring Container

- Container is a software application (or) Java Class that can take care of the whole life cycle of given resource.

- Spring container is nothing but a Java Class which is provided by spring framework but it is not .exe file (or) .bat file (or) not a setup file.



- ✓ Bean Factory can only provides "**Dependency Injection**"
- ✓ Application Context can provide "**Dependency Injection, AOP, I18N,EventHandling**" etc.,
- ✓ BeanFactory & ApplicationContext are interfaces from Spring library.

Spring Container Classes

- BeanFactory(I) and its implementation Classes.
  1) XMLBeanFactory
  2) SimpleJndiBeanFactory
  3) DefaultListableBeanFactory

- ApplicationContext(I) and its implementation Classes.
  1) ClassPathXmlApplicationContext
  2) FileSystemXmlApplicationContext
  3) XMLWebApplicationContext

- Activating the Spring container in spring application is nothing but creating object an implementation class for either of any Spring containers.

- Most frequently used implementation class are "XMLBeanFactory, ClassPathXMlApplicationContext" in spring application development.

- We can't activate the Servlet Container, JSP Container by creating objects for certain classes so they are heavy weight containers these containers will activate during server starup time itself.

Spring Configuration

- Spring Container creates spring bean object, performs life cycle operations, inject dependencies and finally destroys spring bean object.
- Every Spring bean in Spring application has to be configure in **Spring configuration file**.
- Spring configuration file is nothing but an xml file created by programmer/Developer. It should be configured every spring bean in spring application.
- We can provide any file name to Spring configuration file i.e., anyname.xml.
- We can create more than one spring configuration file in spring application.
- Spring configuration File is a simple xml file with beans and their dependencies configuration.

anyname.xml

Spring Bean Configuration

Spring Bean Configuration


```
Class Employee{
   ..............
   ..............
   ..............
}
```

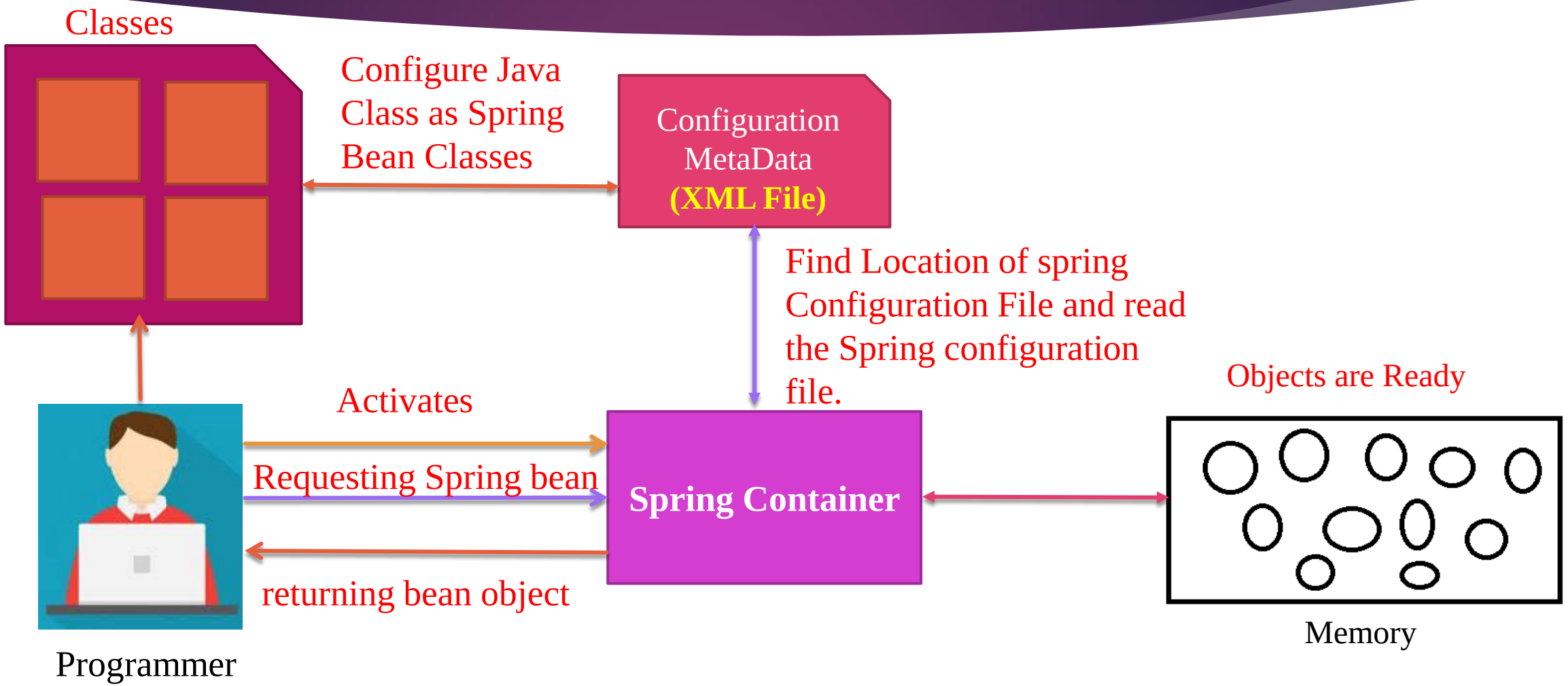
```
Class Customer{
   ..............
   ..............
   ..............
}
```

```
<beans xlmns =".............................
              .............................
              .............................">

   <!-- configuring the Employee class -->
  <bean id="emp" class="com.ait.Employee">
    ....................................
    ....................................
  </bean>

   <!-- configuring the Customer class -->
  <bean id="emp" class="com.ait.Customer">
    ....................................
    ....................................
  </bean>

<beans>
```

Spring Bean

- Spring bean is a key concept of the Spring Framework.
- Spring beans are nothing but Simple Java Class which doesn't extends (or) implements third party related classes & interface.
- Spring beans are always managed by Spring IOC Container mean that Till Object Creation to Object Destruction will be taking care of everything by Spring Framework.
- We can represent the Java Class as Spring Bean below ways
  1) XML Configuration
  2) By Using Stereo Type Annotation(@Component, @Service, @Repository, @Controller)
  3) Java Based Configuration (@Bean)
- Every Spring Bean in Spring framework represented as "Singleton" by default.

Basic Flow



Dependency Injection

- Dependency Injection is a fundamental concept of Spring framework.
- Spring container will inject the dependencies required for class at runtime through Dependency Injection Mechanism.
- Spring framework will provide three kinds of Dependency Injections(**Setter, Constructor, Interface**)

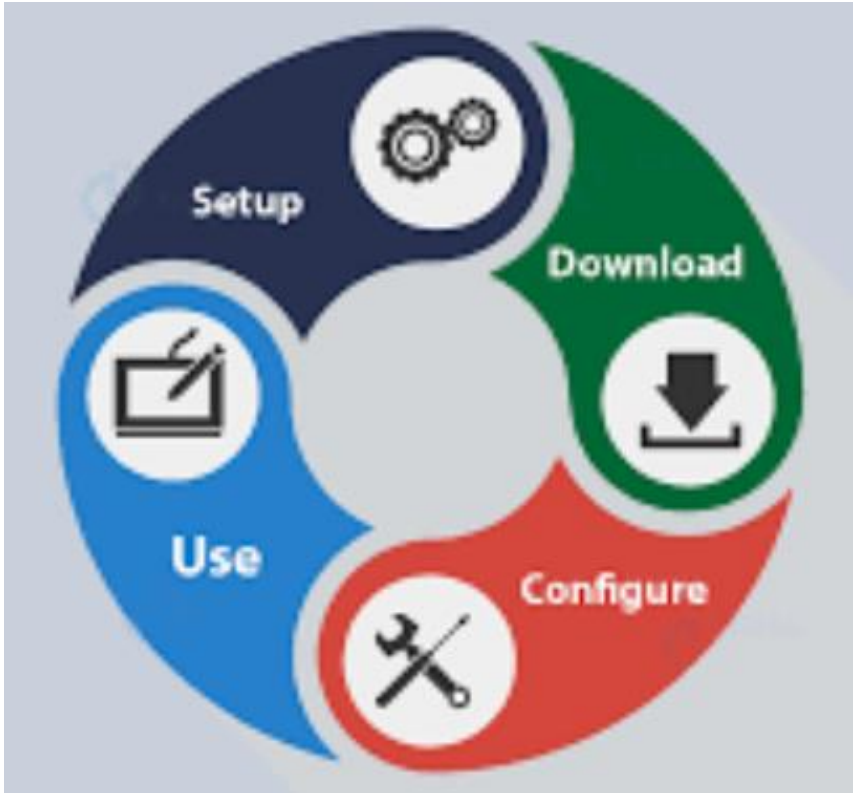
Setter Injection

```
Class Employee{

    private Address address;

    public void setAddress(Address address){
        this.address = address;
    }
}
```

Constructor Injection

```
Class Employee{

    private Address address;

    public Employee(Address address){
        this.address = address;
    }
}
```

Spring Environmental Setup



- ✓ Java Software with Minimum 1.8 Version
- ✓ https://www.eclipse.org/downloads/
  - **Eclipse IDE**
- ✓ https://spring.io/tools
  - **STS IDE**
- ✓ https://repo.spring.io/ui/native/release/org/springframework/spring/
  - **Spring Software Repo**
- ✓ **Maven Build Tool for dependency Management**
- ✓ **More Information About Spring**
  https://spring.io/projects/spring-framework/#learn

Spring First Application

1. Creating Spring Bean Class.
2. Creating Spring Configuration File.
3. Creating Spring Client Class with main method.
   - ❖ Creating Resource(I) Object to hold the spring configuration file.
     - ClassPathResource(IC).....FileSystemResource(IC)
   - ❖ Activates the Spring Container Object by Programmer
   - ❖ Request the Spring Bean Object from Spring container.
   - ❖ Call the Spring Bean Service Methods…

Q & A Session

For More Information : https://ashokitech.com/