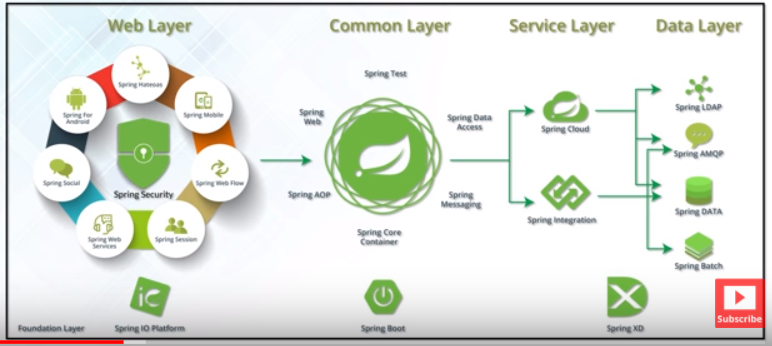
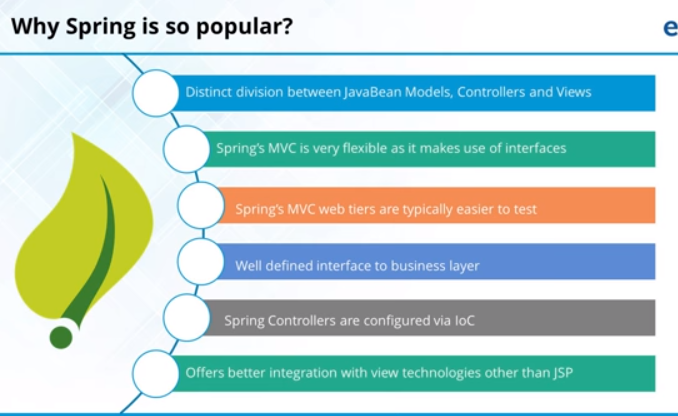
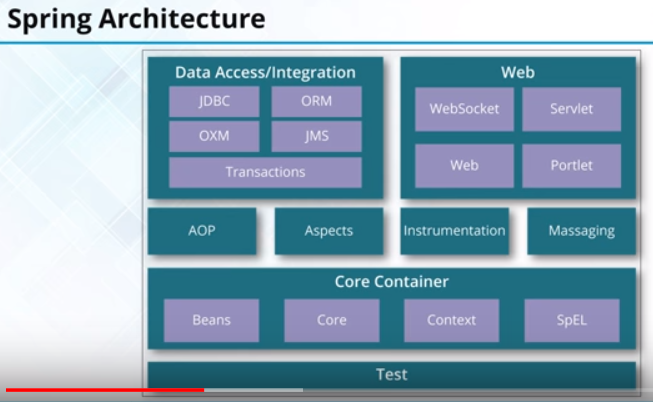
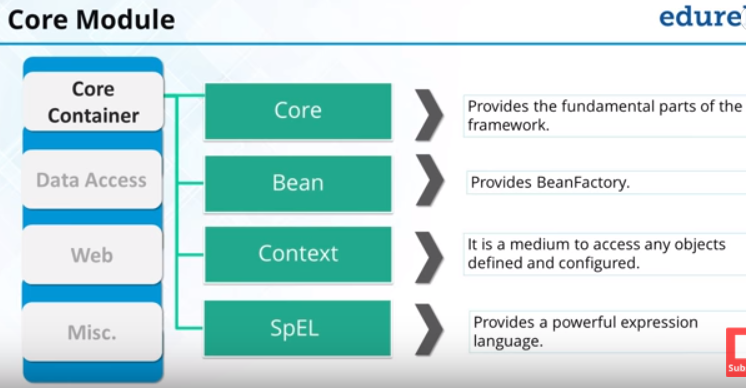
Spring:

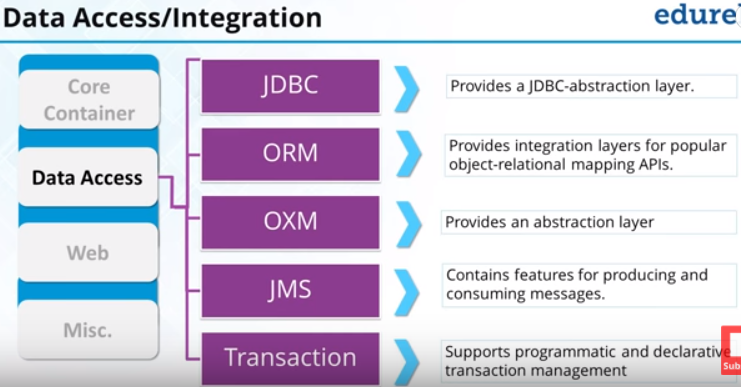
* It has interfaces only so classes can be written without affecting the client side
* 
* Spring is MVC-Model View controller
* 

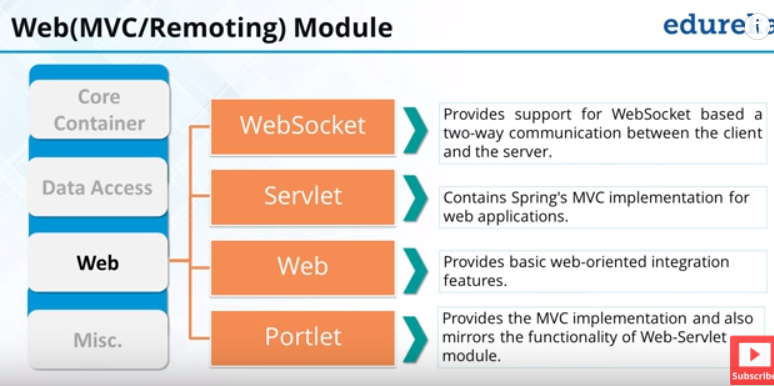


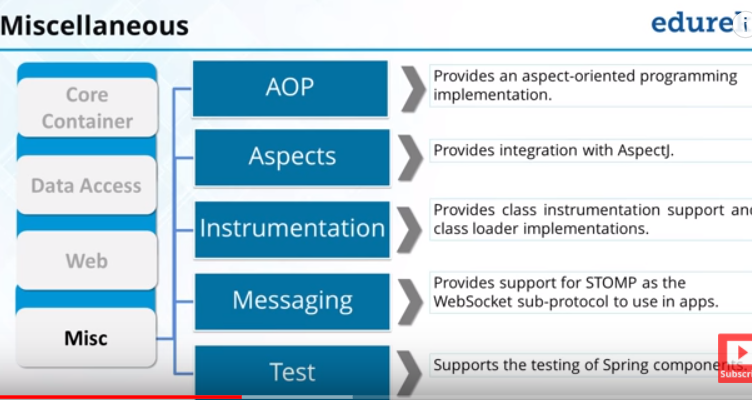
Aspect oriented programming is analogous to OOPS



Spring Expression language-SPEL

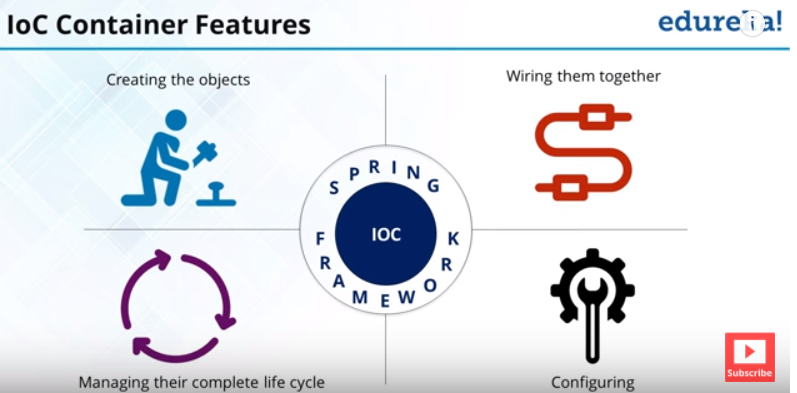






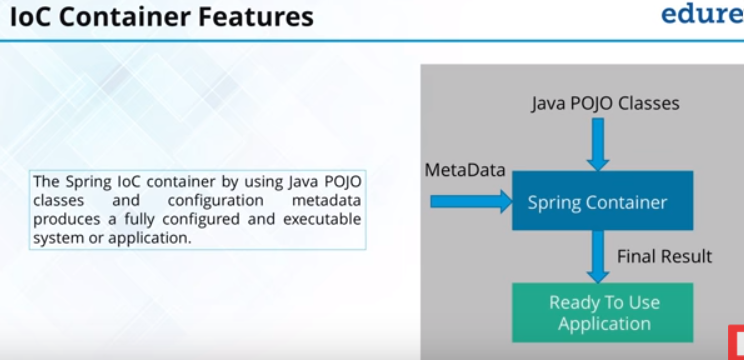
Spring container is responsible for creating object from XML

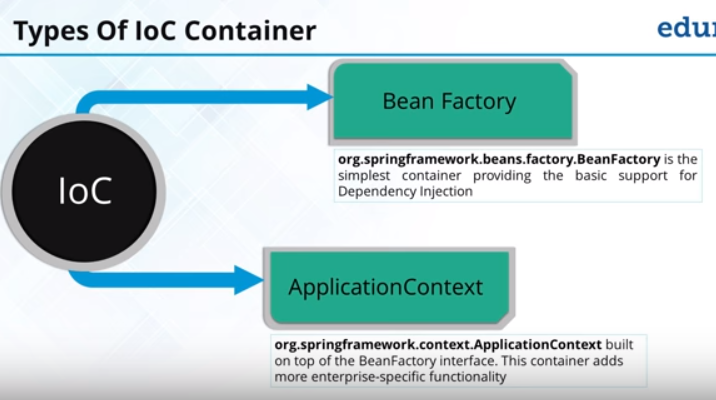
IOC-inversion of control-Spring will create object from XML



Configuring –key value pair

Spring container manage life cycle of object





//Bean container 1--constructor is called during object instantiation

Resource resource=**new** ClassPathResource("employeebean2.xml");

BeanFactory factory=**new** ~~XmlBeanFactory~~(resource);//Spring parser which shall parse the xml File and construct objects for small application BeanFactory is a interface

employee e1=(employee)factory.getBean("emp1");//get baean gives object of objectwhich needs to be typecaseted

employee e2=factory.getBean("emp2", employee.class);

System.***out***.println(e1);

System.out.println(e2);

//Bean Cntainer 2 constructor is called ApplicationContext is a interface context for large application

ApplicationContext context= **new** ClassPathXmlApplicationContext("employeebean2.xml");

employee e3=(employee)context.getBean("emp1"); //context will got to the xml file and get the bean details of the class

// employee e4=context.getBean("emp2", employee.class);

System.***out***.println(e3);

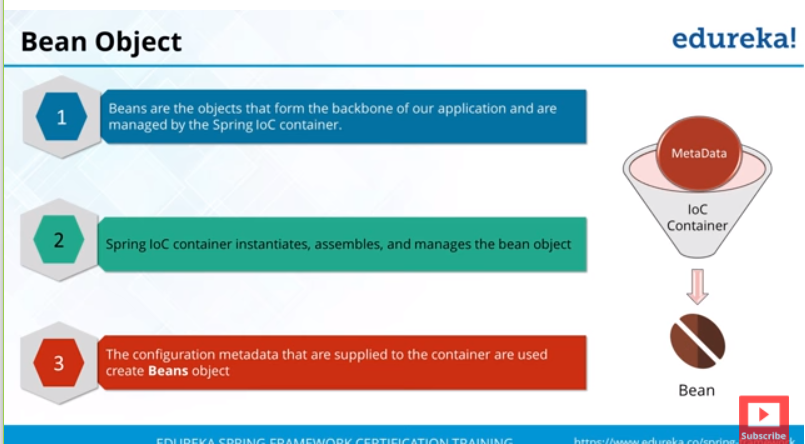
//System.out.println(e4);

Xml:

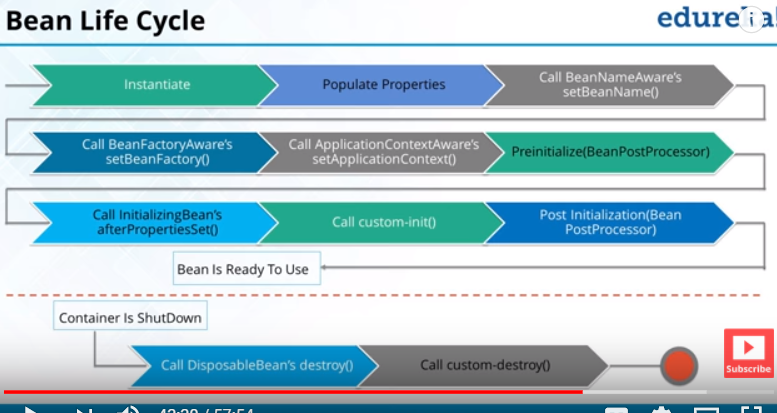
The xml file starts with <beans> and definition is to be given in double quotes and in <bean> tag id and class is given

<bean id="add" class="co.employee.Address" > where id is the id of the bean given in getbean() method and class id the class for wihic the bean is to created

By changing the class name in class file we can create a object which needs change in xml only and no recompilation of source code



Without setter method in bean class IOC is not possible

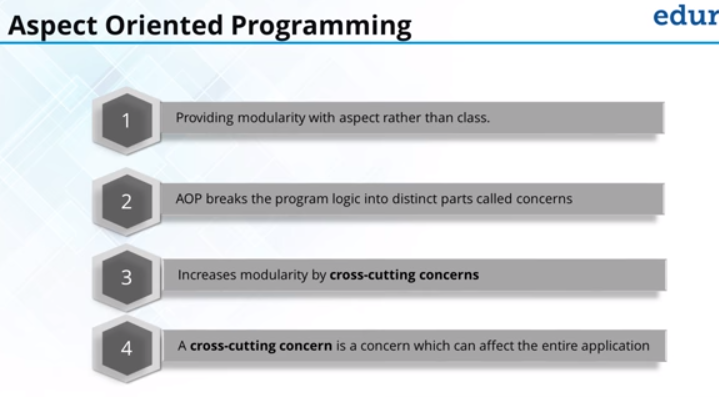


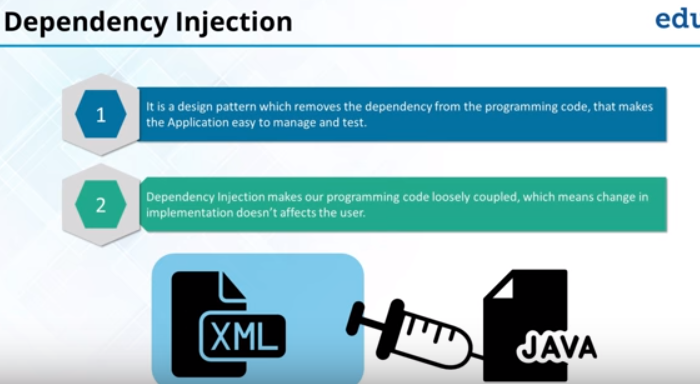
ClassPathXmlApplicationContext cxt=(ClassPathXmlApplicationContext)context;

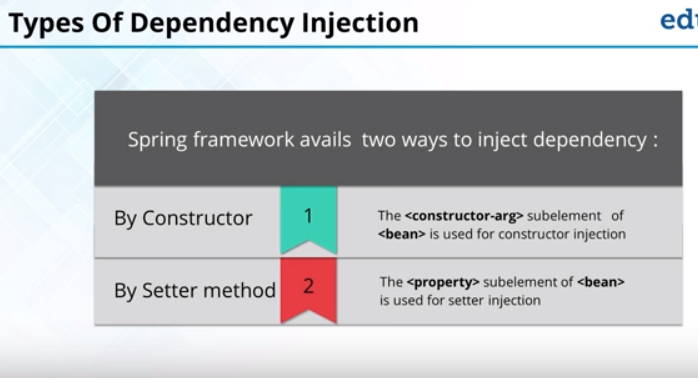
cxt.close();

<bean id="emp1" class="co.employee.employee" init-method="myInit" destroy-method="myDestroy">

Create init and destroy methos in bean







Constructor:

<bean id="add" class="co.employee.Address" >

<property name="city" value="Nellai" ></property>

<property name="State" value="Tamil Nadu" ></property>

<property name="zipcode" value="627006"></property>

</bean>

<bean id="emp1" class="co.employee.employee" init-method="myInit" destroy-method="myDestroy">

<property name="eid" value="102" ></property>

<property name="ename" value="abc" ></property>

<constructor-arg ref="add" /> //to initialize using constructor

</bean>

To initialize with constructor a varalbe:

<constructor-arg value="add" /> //add is passes to the argument of constructor and get assigned

Setter method:

<bean id="add" class="co.employee.Address" >

<property name="city" value="Nellai" ></property> //name of the variable=name,value=value of the variable

<property name="State" value="Tamil Nadu" ></property>

<property name="zipcode" value="627006"></property>

</bean>

<bean id="emp1" class="co.employee.employee" init-method="myInit" destroy-method="myDestroy">

<property name="eid" value="102" ></property>

<property name="ename" value="abc" ></property>

<!--constructor-arg ref="add" /-->

<property name="eaddress" ref="add" ></property>

3 types of configuration for beans:

* Xml based configuration
* Annotation based configuration
  + (@component present in stereotype and the default name will be the class name)
  + In the xml change <context:component scan basepackage=”package name”>and comment the bean tag
* Java based configuration

Auto wiring is used to inject dependency

For annotation based configuration,we need config class and they are annotated with @configuraion -to explain that the class is responsible to give the objects

@bean-is used to specify the bean class

@componentscan-is used for creating beans without auto configuration class.Hust specify the class name

@component-to specify which bean is to create objects and the objects will be created in spring container for the object during spring args

When 2 classes implement the same interface error,will occur.so choose Qualifier and give class nameS

Single ton bean creation-don’t need to wait for object creation,when spring args is called object is created without get bean(we cannot create two objects

@scope(prototype)-will be used how many times we are instatiating the object(using get bean)-will not create a object on spring args

**public** **class** samsungclass {

@Autowired //check in configuration

mobileprocessor cpu;

Samsung class is dependent on mobileprocessor class,to make samsungclass to aware of the dependency we use autowire

@autowire will search for the object in spring container

@component(“anme1”) –will create a object with anme1

@qualifier(“anme1”)-will search in container with anme1

The @Configuration annotation indicates that the class is a source of bean definitions. Also, we can add it to multiple configuration classes.

The @Bean annotation is used on a method to define a bean. If we don’t specify a custom name, the bean name will default to the method name.

Constructor based annotation:

@Configuration

public class AppConfig {

    @Bean

    public Item item1() {

        return new ItemImpl1();

    }

    @Bean

    public Store store() {

        return new Store(item1());

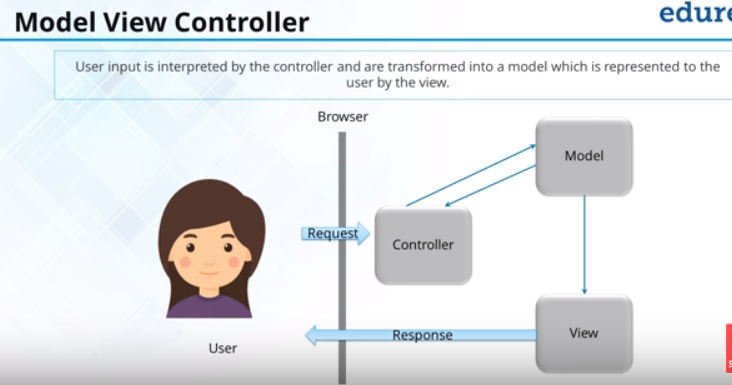
    }

}

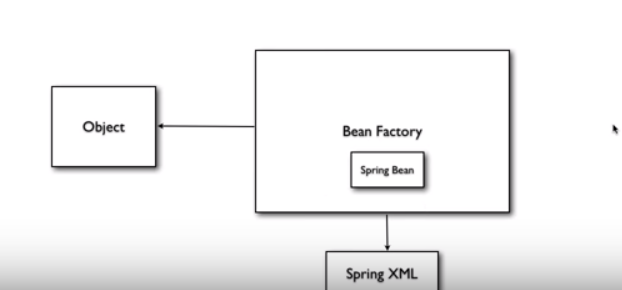
**Setter-Based Dependency Injection**

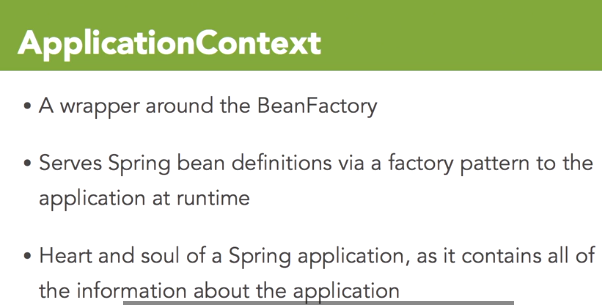
For setter-based DI, the container will call setter methods of our class, after invoking a no-argument constructor or no-argument static factory method to instantiate the bean. Let’s create this configuration using annotations:

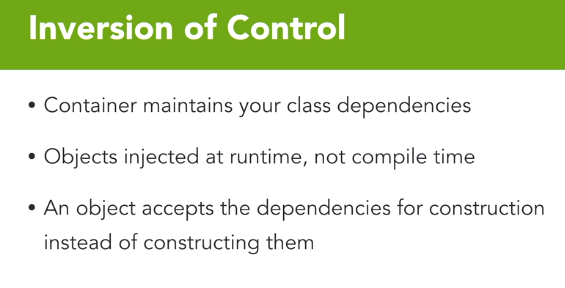
|  |  |
| --- | --- |
| 1  2  3  4  5  6 | @Bean  public Store store() {      Store store = new Store();      store.setItem(item1());      return store;  } |

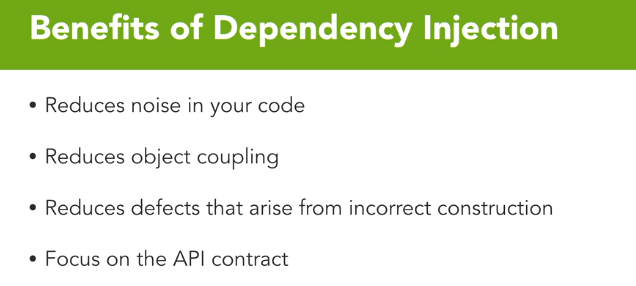


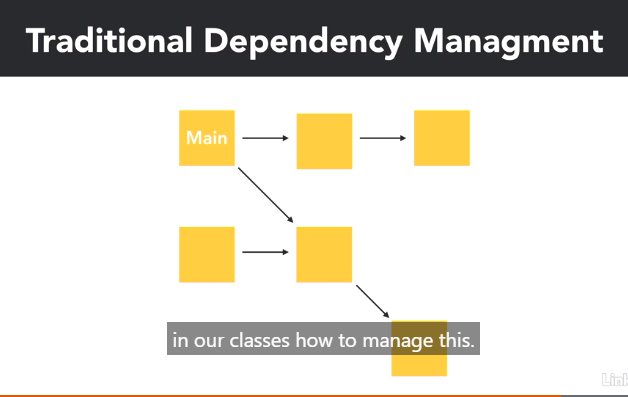
A spring container has beans,handles the object lifecycle

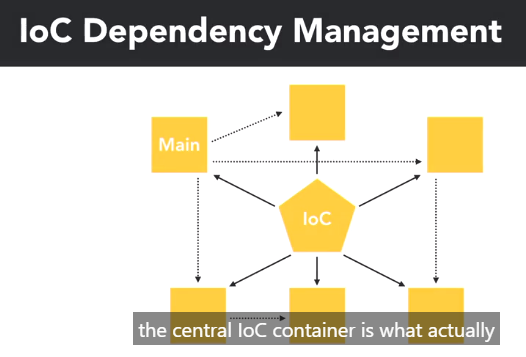




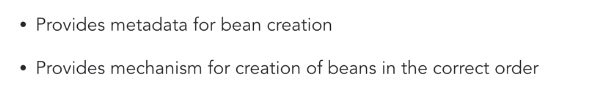


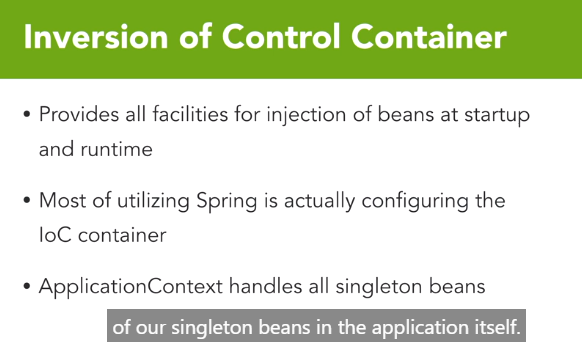


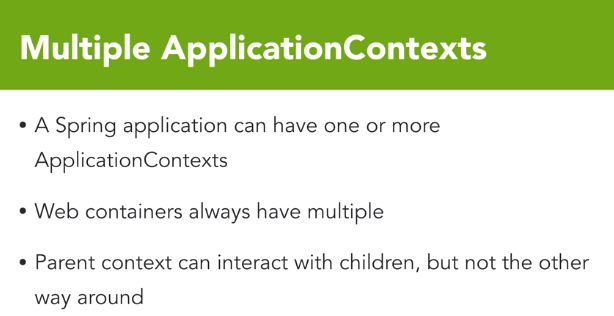


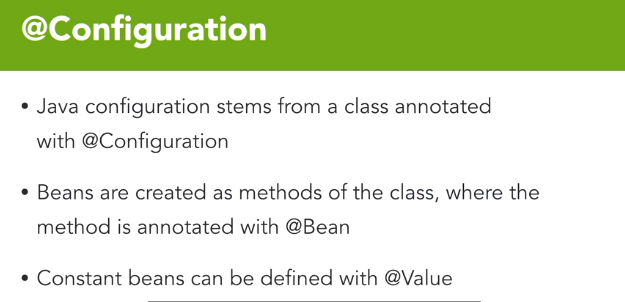












In spring the environment variable gn at runtime is taken as priority compared to properties value

To create different profile:

@Bean

@Profile("dev")

**public** Worker workerForDev(){

**return** **new** Worker("Hello", greetingText);

}

@Bean

@Profile("prod")

**public** Worker workerForProd(){

**return** **new** Worker("Greetings", greetingText);

}

Run with environment variable spring.profiles.active value as dev or prod

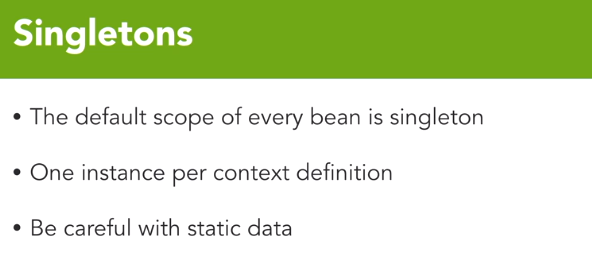
Spring Expression:

For changing profile using spring expression

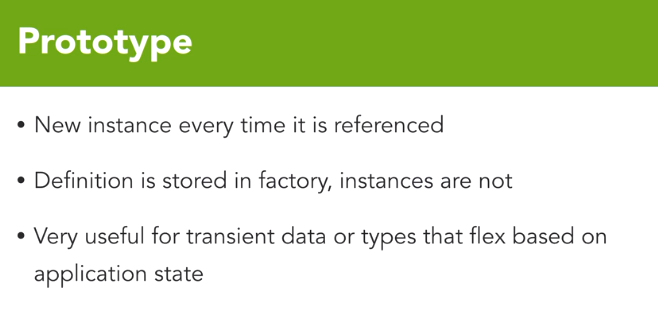
spring.profiles.active –changed in runtime environment variable

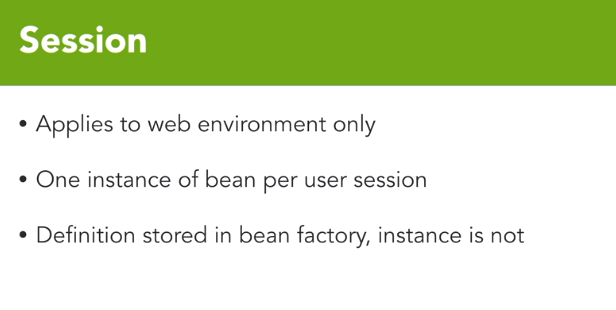
create 3 application properties

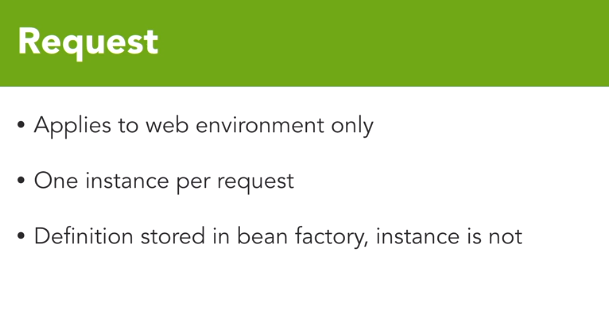
@PropertySource("classpath:/application-${spring.profiles.active}.properties")



Prototype bean is available for garbage collection while single ton is not







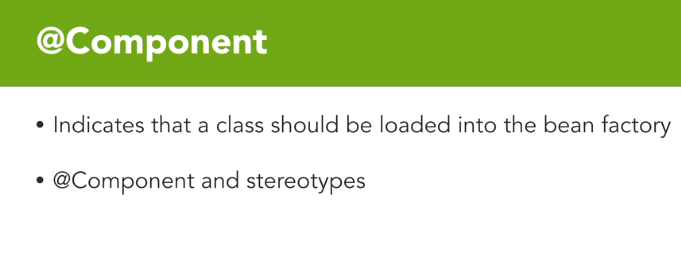
Proxies:

* Is based on aspecting
* Are aspects which add various behavior to your class
* The common proxies are transaction management and caching
* Every bean adds one proxy



Annotation based configuration:

* This is another way of configuring application context
* Other stereotypes include @controller,@service and @repository
* Dependency injection is achieved through autowiring
* @qualifier is used when multiple implementation of interface are needed

=



IOC container:

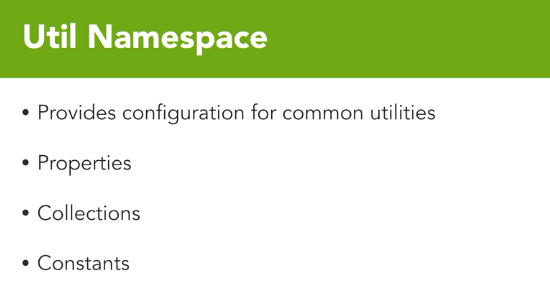
Is responsibe for dependency management

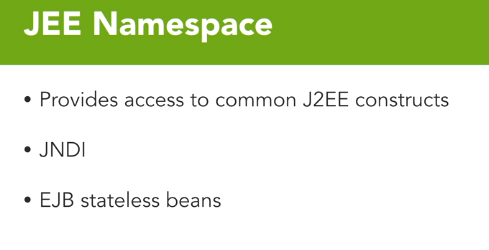
BeanLifecycle:

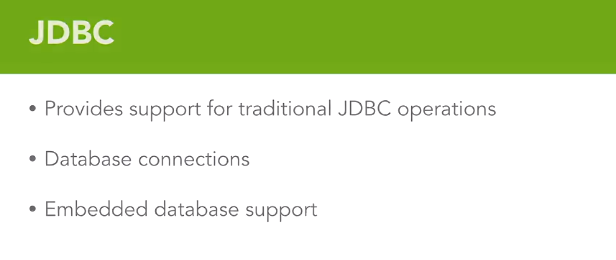
* @postconstruct is used for destructor and can be used for only one method

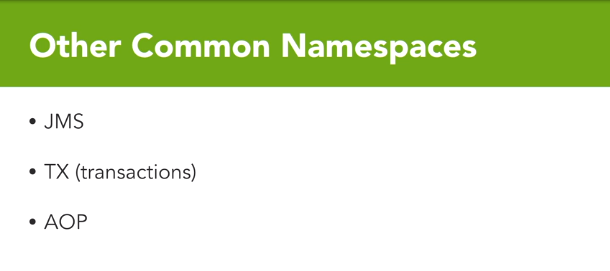
XML configuration

* It has 2 parts
  + Bean configuration
  + XML namespaces used





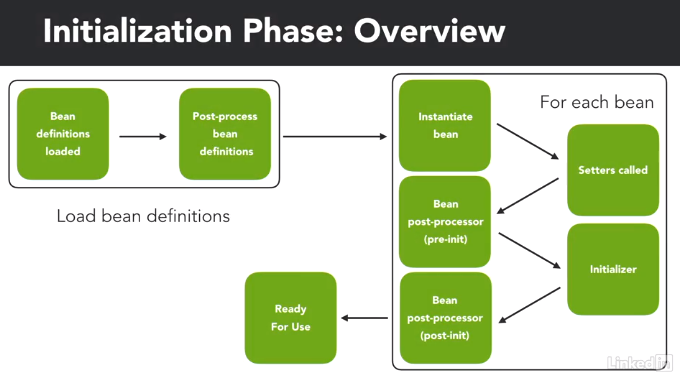




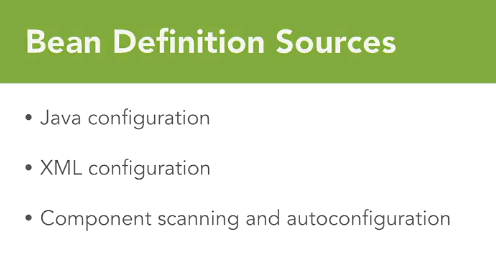


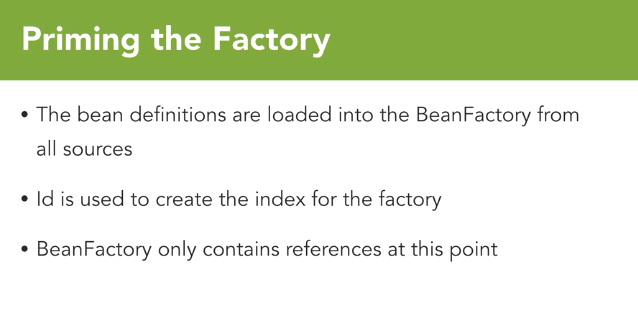
Initialization-Application context

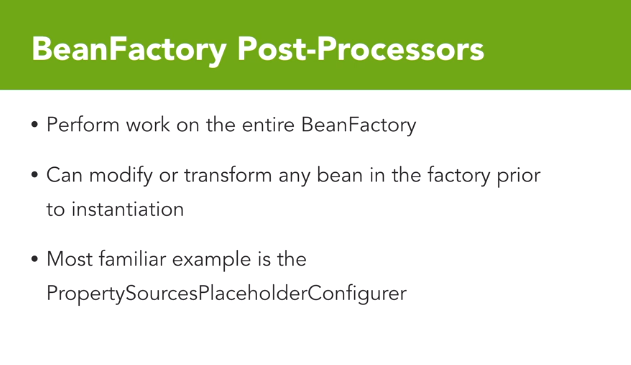
Application context is wrapper for beanfactory

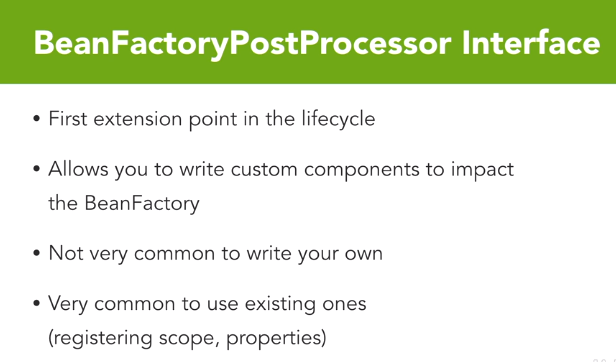


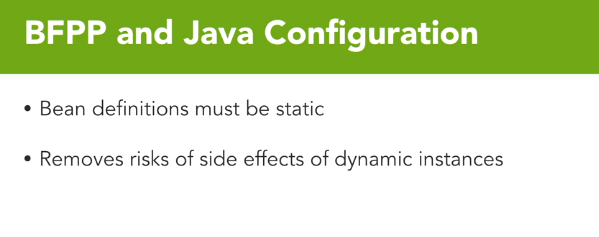
Loading the beans:

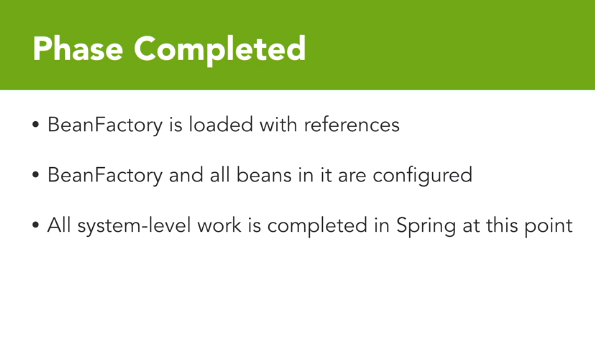


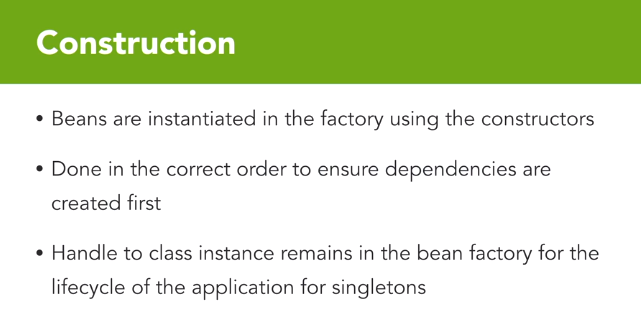


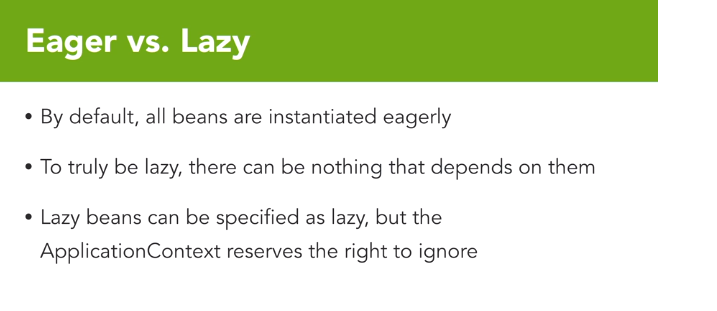


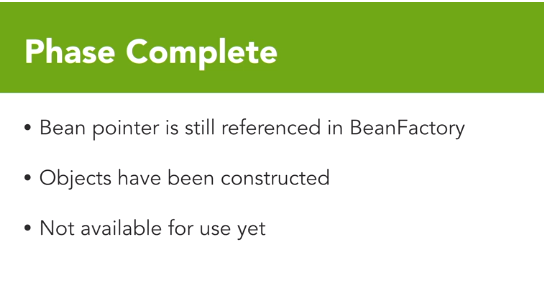


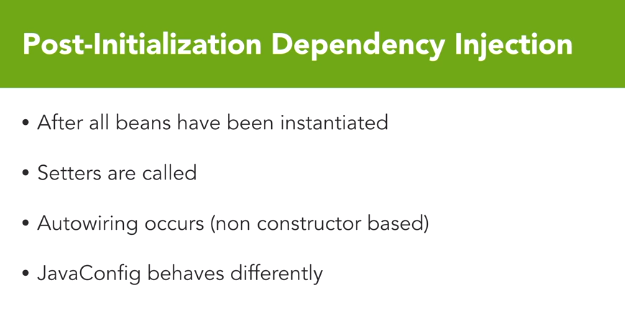


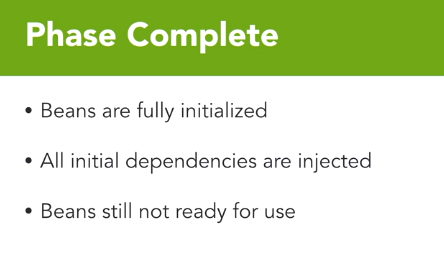




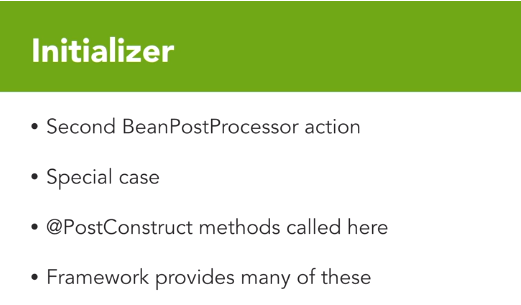


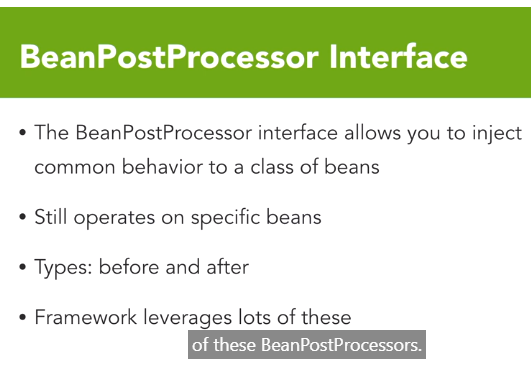


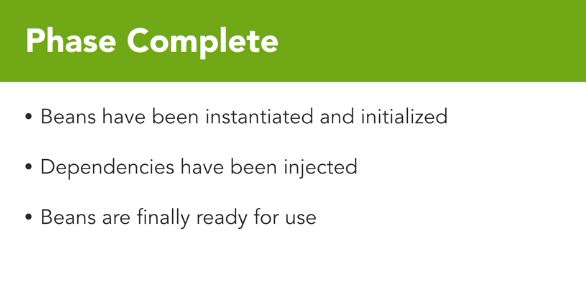


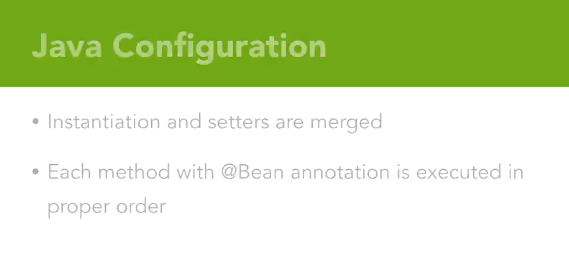


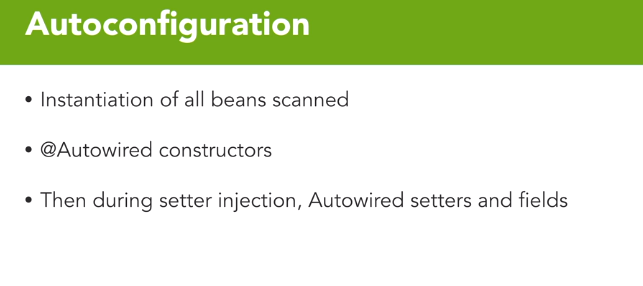


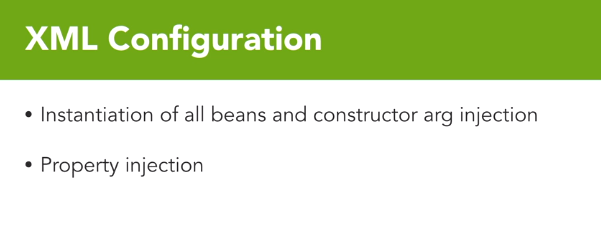


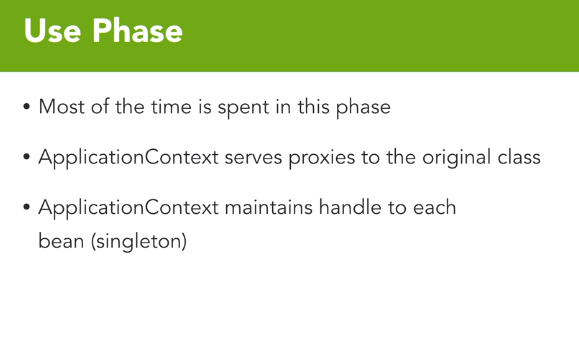


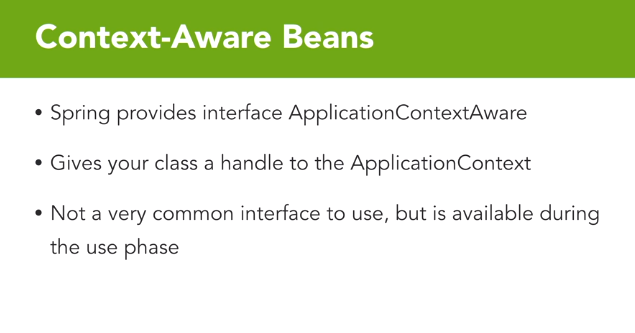


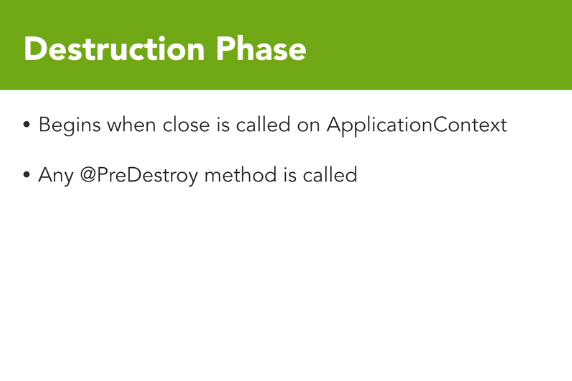


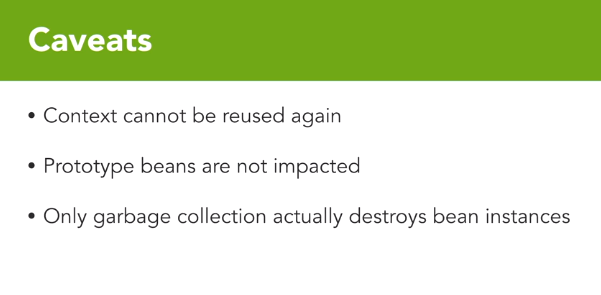


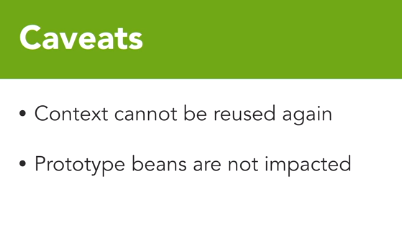


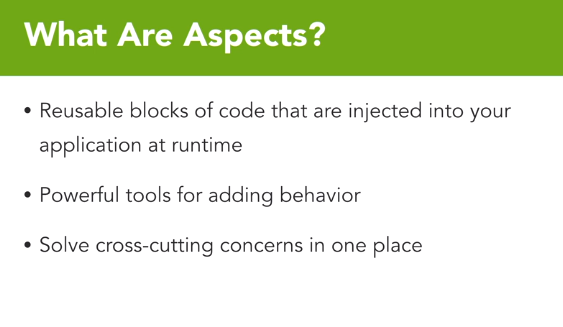




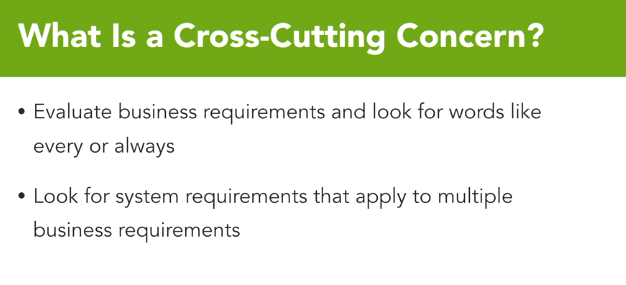


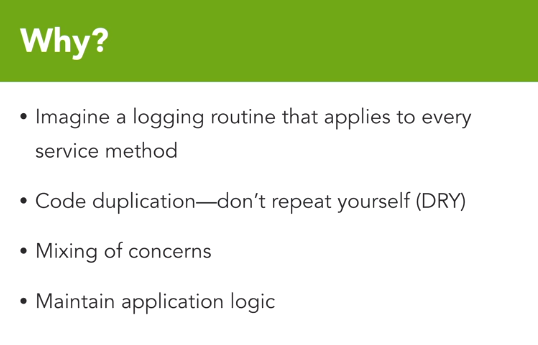


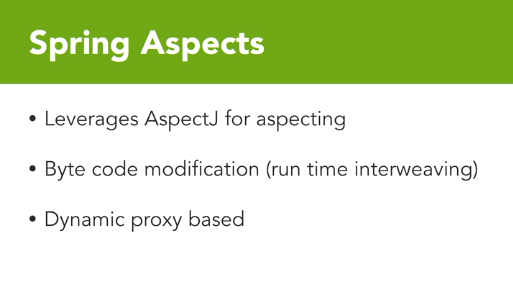


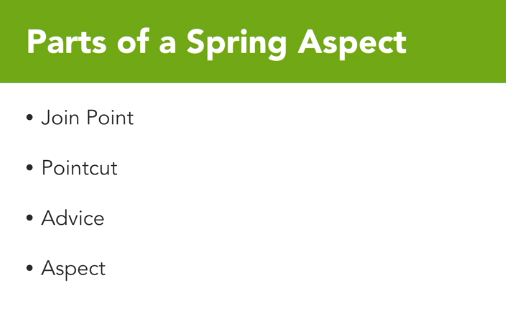


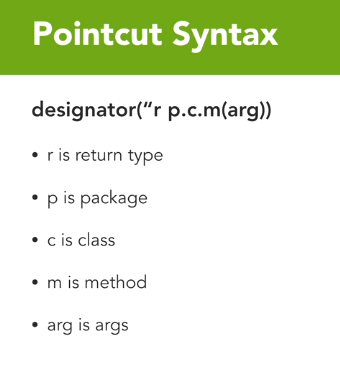


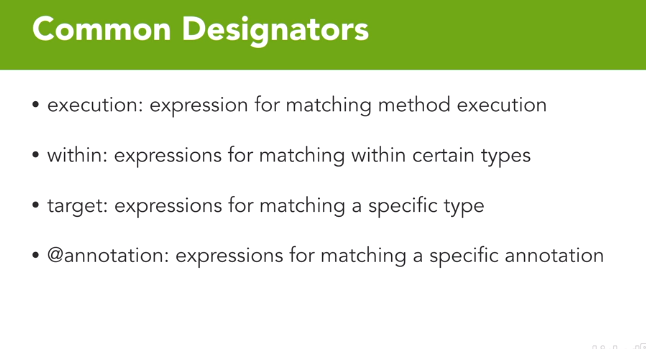












• 4 years of experience in database and automation testing expertise in e-commerce domain.

• Well-versed in creating Test plans, identifying Test Scenarios, designing Test Cases and execution of designed test cases.

• Experienced in Functional testing, Smoke Testing, Regression Testing.WebServices Testing, Defect logging and Defect tracking activities,Agile Methodology.

• Worked closely with other testers and developers for bug fixes and feature requests.

• Worked on JIRA ,QTest.Good technical and analytical skills.

• Developed automation tools using selenium

• Worked on Websphere commerce(WCS),Order Management sytems(OMS),Sterling Order Management System

• Basic knowledge in Java. SQL language and data structures.

• Had undergone Java training in Global Education Centre,Infosys.

• BTech Information Technology from MCET

Keen to be a part of an organization where my professional experience & academic qualification can be synergized with the organizational goals.

* 4 years of experience in database and automation testing in e-commerce retail domain.
* Experienced in developing in automation tools using Selenium and Spring in Java
* Worked closely with other testers and developers for bug fixes and feature requests.
* Worked on JIRA,Websphere commerce(WCS),Order Management sytems(OMS),Sterling Order Management System
* Good technical and analytical skills.
* Good knowledge in Core Java,SQL,Spring,Selenium and Data Structures and Algorithms
* Experienced in Functional testing, Smoke Testing, Regression Testing.WebServices Testing, Defect logging and Defect tracking activities,Agile Methodology
* Had undergone Java training in Global Education Centre,Infosys.
* BTech Information Technology from MCET

Keen to be a part of an organization where my professional experience & academic qualification can be synergized with the organizational goals.

Experienced in Functional testing, Smoke Testing, Regression Testing.WebServices Testing, Defect logging and Defect tracking activities,Agile Methodology.

• • Worked on JIRA ,QTest.Good technical and analytical skills.

• Developed automation tools using selenium

• Worked on Websphere commerce(WCS),Order Management sytems(OMS),Sterling Order Management System

• Basic knowledge in Java. SQL language and data structures.

• Had undergone Java training in Global Education Centre,Infosys.

• BTech Information Technology from MCET

* Creating automation tools for order processing using Spring and Selenium framework
* Validation of different Post Order flows like Order release, shipment, cancel and return.Email testing, financial processing and EOD report validation.
* Testing for different the Fulfillment types including ship and pickup for orders created from different channels like online and POS.
* Validating and verifying data integrity and performed backend data validation using MySQL,service and log validation
* Execution of automation scripts
* Coordination with Development team to track the defects to closure.
* Effective Offshore Onsite Coordination and reporting.
* Good knowledge in WCS, OMS , SQL,Sterling OMS
* Creating Test plans, identifying Test Scenarios, designing Test Cases and execution of designed test cases,status reporting,Defect tracking and mangement
* Worked as an individual contributor being a part of QA team

-Validation of orders from online reaches the OMS without any data discrepancies by verifying the CO XML values passed from CRSP layer is similar to the WCS layer by using REST API

-Automation of order placement process in CRSP ans WCS layer using Selenium

-Automation Testing of discrepancies in Create Order XML from CRSP layer and WCS layer using REST API

-Defect tracking and Management