**Spring Boot**

**Why will you choose Spring Boot over Spring Framework?**

Spring Boot is module of Spring Framework which will speed up the development process.

Spring Boot is nothing but a Spring Framework for Rapid Application Development with extra support of configuration and embedded server.

Challenges in Spring

1. We to remember all of the dependency which we need to add.
2. Also, we need the manage version for all of this.
3. We need to add so many configurations for creating simple Web App.

It Provide automatic configuration for dependency.

* Dependency Resolution and Version Conflict
* Avoid additional configuration
* Embedded Tomcat, Jetty
* Provide Production ready features such as metrics, health checks

**What is RAD Rapid Application Development means?**

Rapid application development is modified Waterfall model which focus on development application in short span of Time.

Basically there is five phase of development in RAD

**Business Model** – In this we need to create business model for the product.

**Data Model** – After creating or understand business model we need to design data model. Where we create data objects and establishing relation in this data objects.

**Process Model** – with the help of both business model and Data model we need to process on this object like Adding, deleting, retrieving, Modifying.

**Application Generation** – Here we create actual Product prototype and ask the client to Agree on this build code project or not then we will take next step.

**Testing and Turnover** – Now we can test over product on client side if it works properly then ok If changes are required from client side the start over again whole processes known as Turnover.

**What all Spring Boot Starter you have used or what all modules you have worked on?**

Spring Boot Starter Web

Spring Boot Starter Data JPA

Spring Boot Starter AOP

Spring Boot Starter Web Service

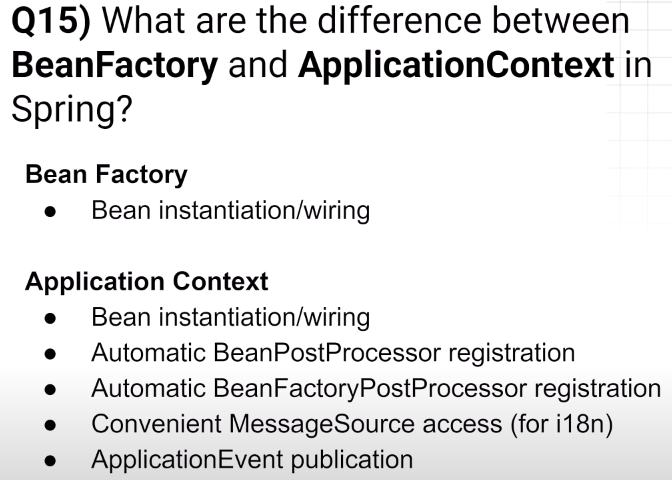
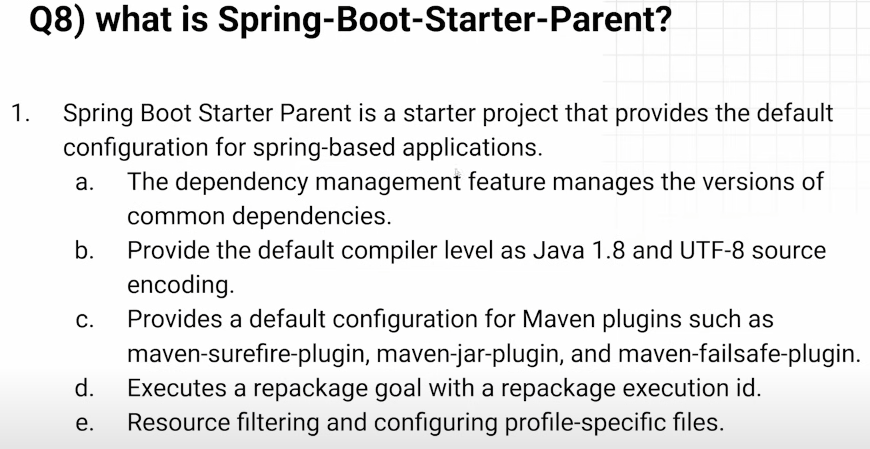
Spring Boot Starter Security

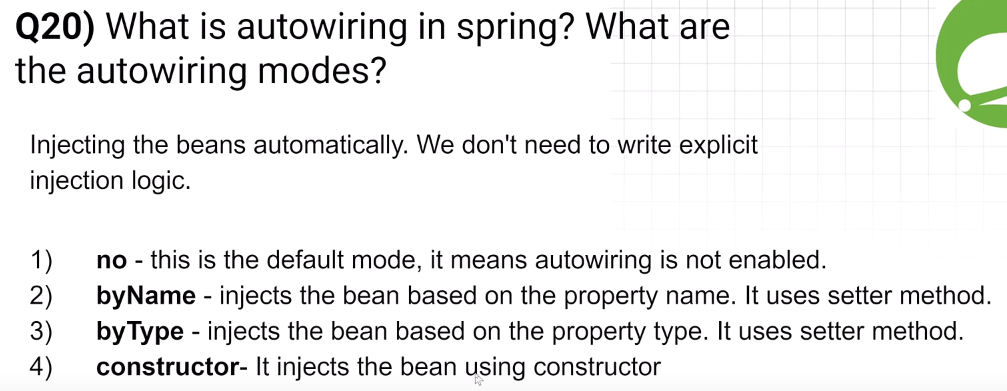
Spring Boot Starter Apache Kafka

Spring Boot Starter Spring Cloud

**What is Spring Boot Starter?**

Starter are collections of pre-configured dependencies that make it easier to develop particular kinds of application.





**How will you run your spring boot application?**

**What is Auto Configuration in Spring Boot?**

If we want to know how many things will Auto configure the n add property “debug=true”.

**How can you disable a specific auto-configuration class in Spring Boot?**

@Configuration(exclude = “SomeExcludeClass.class”)

OR

@SpringBootApplication(exclude = “SomeExcludeClass.class”)

**What is Dependency Injection?**

First the dependency injection is a design pattern and it is use loose coupling between the classes.

**Where to use setter injection over constructor injection and vice versa?**

**How many ways to perform dependency injection in spring or spring boot?**

1. Field Level DI
2. Setter DI
3. Constructor DI

Now First Constructor injection we have go for this

**When you want your dependency is mandatory for your bean means it must need to inject at run time.**

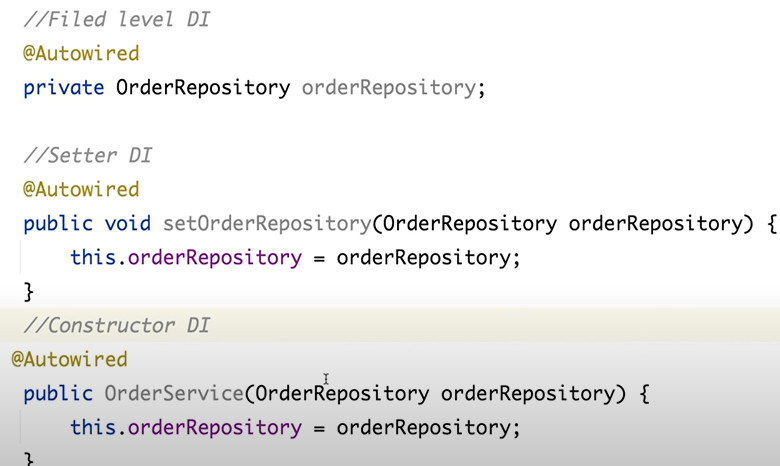
**It is Immutable in nature**

It ensure that all the mandatory dependency injection is done at run time.

Now Setter injection

**If dependency is optional then we go with setter injection.**

**It is mutable in nature.**

****

**Where you would choose to use setter injection over constructor injection, and vice versa?**

If we must need to inject dependency at run time then we need add constructor dependency injection.

Setter dependency injection like lazy loading so we will use when we need this dependency means optional dependency.

Also to avoid circular dependency while using constructor dependency injection then we use setter injection.

Circular or cycle dependency injection means simple A class have constructer dependency injection in B class and B class have constructor dependency injection in A class.

**Can you provide an example of a real world use case where @PostConstruct is particularly useful.**

Using @PostConstructor annotation we will define pre-processing logic like Kafka producer and consumer or database logic.

**How can we dynamically load values in a Spring Boot application?**

Using @Value.

**Can you explain the key differences between yaml and properties file, and in what scenario you might prefer one format over the other?**

**What is the difference between yml & yaml?**

In olde version of windows there is restriction of three character extension that’s why we use yml.

Now we can use both.

**If I will configure same values in both properties then which value will be loaded in Spring Boot or who will load first properties or yml file?**

Properties file first.

**How to load external properties in Spring Boot?**

spring.config.import

**How map or bind config properties to java object?**

Using @ConfigurationProperties(prefix=“spring.datasource”)

**How will you resolve bean dependency ambiguity?**

Using @Qulifier

**Can we avoid this dependency ambiguity without using @Qualifier?**

Yes using @Resouce(name = “someClassName”)

What is bean scope and can you explain different type of bean scope?

**How to define custom bean scope?**

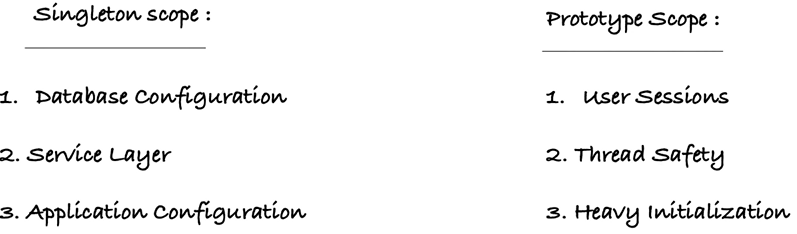
1. Implement Scope interface.
2. Overrirde all methods
3. Now we need to register beanFactory

Scope scope = new CoustomeThreadScope();

Context.getBeanFactory().registerScope(“threadScope”,scope);

1. Now annoted that CoustomeThreadScope class which is implemented by Scope interface.

**Can you provide a few real-time use cases for when to choose Singleton scope and Prototype scope**



**Can we inject prototype bean in singleton bean? if yes what will happen if we inject prototype bean in singleton bean?**

**What is difference between spring singleton and plain singleton?**

**What is the purpose of the BeanPostProcessor interface in Spring, and how can you use it to customize bean initialization and destruction?**

**How can you customize the default configuration in Spring Boot?**

**OR**

**How we can change the server port no. in Spring Boot?**

In application.propertys file we can write server.port=8081.

**How Spring Boot run() method work Internally?**

* Create ApplicationContext
* Register bean into context
* Kicked up embedded tomcat container to run our Jar / War file.

**Can we replace the embed tomcat server in spring boot application?**

Yes, with adding the dependency like **spring-boot-starter-jetty**.

**Can we disable web server in spring boot application?**

Yes, we can write in application.property file “**spring.main.web-application-type=none**”.

**How to disable Auto-configuration class?**

We can add @EnableAutoConfiguration(exclude=”excudingclasssname.class”).

**Explain the purpose of Stereotype annotations in Spring Boot?**

First of All Stereotype Annotations are used to indicate the role or purpose of a class within a spring application.

@Component – this annotation is parent class and all other will inherit from Component

@Service – all the business logic will comes here

@Repository – DAO layer logic will comes here

@Controller – All the web specific logic comes here

@RestController

**Can we use @Service in repository class and @Controller in Service class and so on?**

Yes we can use but it cannot have any logic to do.

**How can you define bean in spring framework?**

By defining Stereotype annotation spring will create beans for application.

For manual we can do like

We need to create configuration class with @Configuration annotation

In this class we need to define bean

@Configuration

class AppConfig

{

@Bean

public DemoSevice demoService()

{

return new DemoService();

}

}

**Have you worked on Restful web services? If yes what all HTTP methods have you used in your project?**

PUT

POST

GET

DELETE

**How can you specify the HTTP method type for your REST endpoint?**

**Can you design a rest endpoint, Assume that you have a product database, and your task is to create an API to filter a list of products by productType.**

**Design endpoints in a way that takes "productType" as input. If the user provides this input, the endpoint should filter products based on the specified condition. If "productType" is not provided, the endpoint should return all the products.**

**What is the difference between @PathVariable and @RequestParam?**

PathVariable – it take variable from given url

@GetMapping(“/search/{userID}”)

public String getUserNameByID()

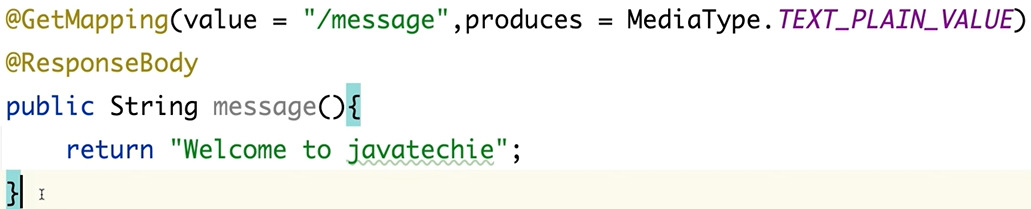
when request input is optional, we will go with the @RequestParam

**Why did you use @RestController and why not @Controller?**

**@Controller –** Spring framework will always be looking for HTML or view page for the returning string from method.

If we annoted @Controller it will always expect as return type Model and View

**@RestController** – It will return any thing we have in our method.



**How can we deserialize a JSON request payload into an object within a Spring MVC controller?**

Using **@RequestBody** annotation we are instructing to the spring to deserialize the incoming JSON request to the particular object

**Can we perform update operation in POST http method if yes then why do we need PUT mapping or put http method?**

Yes we can do it but its violet the rules of HTTP method.

PostMapping is create the new resource.

PutMapping is use to update the existing resource.

**Can we pass Request Body in GET HTTP method?**

Yes we can do but it is not best practice code.

**How can we perform content negotiation (XML/JSON) in Rest endpoint?**

Content Negotiation mean it will return different media type.

Just need to add in annotation attribute “produces”

For Example

@GetMapping(produces = {“application/json”,”application/xml”})

public List<Product> getProducts()

{

return productService.getAllProducts();

}

For XML we need to add dependency in pom.xml which is jackson-dataformat-xml.

**What all status code you have observed in your application?**

**How can you customize the status code for your endpoint?**

Just add @ResponseStatus annotation.

**@ResponseStatus(HttpStatus.*CEATED*)**

**@PostConstruct – use for pre-processing logic**

When we want to perform some specific code at the of application start-up then we can use this annotation.

**Can you explain deference between YML and properties file and which scenario you might prefer that?**

YML is more readable in complex configuration and in structured way.

In properties file it does not follow any hierarchy it just plain line we write it any order.

In YML reduce prefix.

In YML it support list and arrays.

**How can you enable cross origin?**

Cross origin is a security future use for controlling web page request from one domain to another domain.

@RestController

public class GreetingController {

private static final String template = "Hello, %s!";

private final AtomicLong counter = new AtomicLong();

@CrossOrigin(origins = "http://localhost:9000")

@GetMapping("/greeting")

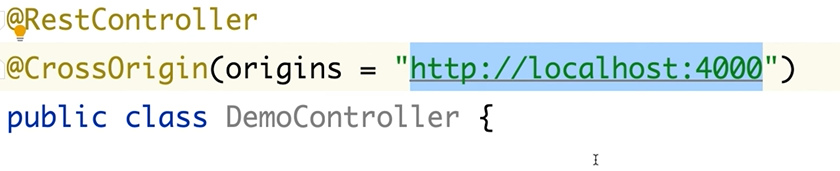
public Greeting greeting(@RequestParam(required = false, defaultValue = "World") String name) {

System.out.println("==== get greeting ====");

return new Greeting(counter.incrementAndGet(), String.format(template, name));

}

}





**How can you upload a file in spring?**

**How do you maintain versioning for your REST API?**

**How will you document your REST API?**

Using Swagger now its OpenApi-starter-webmvc-ui

**How can you hide certain REST endpoints to prevent them from being exposed externally?**

**How will you consume restful API?**



**@ControllerAdvice**

This annotation is use for to handle all exceptions globally.

It applies globally to all the controller whenever ever any exception occurs.

**@ExceptionHandler**

This annotation handles the specific Exception and send custom responses back to client.

**How will you handle exceptions in your project?**

* First we create a package which will have exception handling class.
* Now we create a class for a custom exception.
* Now in this class we can extend RunTimeException and create constructor with msg.
* Now we will call this class where the method might be throw this exception with the specific msg.
* Now we will create another Advice package which will handle rest api exception modification.
* For this we need to create class Like SomeExceptionHandler and annoted with @RestControllerAdvice.
* In this class we can create a method which will modify the rest api exception in proper msg.
* For this created class like name as errorHandlermodifiy which having fields like status, errorMessage, StatusCode.
* Now this call is return type of our handler method in handler class.
* Add @ExceptionHandlet(whith\_class\_name\_will\_throw\_this\_exception.class).
* And create modified field and return this from this handler method.

**How can you avoid defining handlers for multiple exceptions, or what is the best practise for handling exceptions?**

**How will you validate or sanitize your input payload?**

Add spring-boot-starter-validation dependency in pom.xml.

Now go to this particular bean class which we have to make validation.

Add @NotNull, @NotEmpty, @Min, @Max and so on annotation in field which we have to validate.

For example

If there is field like productPrice and this field have int value and this should be some min value and max value.

So we will add @Min and @Max annotation.

@Min(value = 499, message = “Price should not be less than 499”)

@Max(value = 1000, message = “Price should not be more than 1000”)

How can you populate validation error message to the end user?

**How can you define custom bean validation?**

**use case: lets say you find a bug in production environment and now you want to debug the scenario, how can you do that from your local?**

**How can you enable a specific environment without using profiles? OR what is the alternative to profiles to achieving same use case?**

We will add property in application.propertis file

Spring.profiles.active = prod

**What is the difference between @Profile and @Conditional?**

**What is AOP?**

We can do using AOP each and every secondary logic we can define as an aspect where we can write this transaction related or logging, validation or notification and so on which secondary logic and we can tell to spring here we can use this kind of logic will use.

**Aspect** is nothing a methodology or a class who contains the secondary logic.

What is pointcut & join points in AOP?

What are different type of advice?

use case - Can I use AOP of evaluate performance of a method or is this possible to design a logging framework to capture request and response body of a method?

**How does your application interact with the database and which framework are you using?**

Add dependency

Add important data source properties in application.properties file.

In this file we have **Dialect – dialect will help JPA to generate a query behalf of us.**

Create POJO class and add some annotations.

Now create repository interface and extends JPA repository.

**Why is it important to configure a physical naming strategy?**

* It will create table name like if we have field firstName so JPA repo will create this field in database as column name as a first\_name it will add some underscore in between.
* So now we will configuring this to just adding property in property file
* **“spring.jap.hibernate.naming.physical-strategy = org.hibernate.boot.model.naming.PhysicalNamingStartegyStandardImpl”**

**What are the key benefits of using spring data JPA ?**

It simplifies the development of data access layer by providing the repository as high level abstraction over jpa entity.

So we just create entity class and create corresponding repository and extend JpaRepository and provide class object name and primary column data type field.

**What are the differences between hibernate JPA and Spring Data JPA ?**

**How can you connect multiple databases or data sources in a single application ?**

**What are the different ways to define custom queries in spring Data JPA ?**

**How will you define entity relationships or association mapping in spring Data JPA ?**

**Is this possible to execute join query in spring data JPA ? if yes, how can you add some insights ?**

**How will you implement pagination and sorting in Spring Data JPA ?**

**If same configuration write on both YML and Properties file then which value load first?**

Properties file will load first.

**How load External Properties in spring boot?**

spring.config.import = external file path and name. It will load this value form file.

**How to map or bind config properties to Java Objects?**

Write configuration class with @Configuration annotation.

Then add @ConfigurationProperties() annotation on this class.

Add “prefix” attribute in it @ConfigurationProperties(prefix = “spring.datasource”)

**@Configuartion**

**@ConfigurationProperties(prefix = “spring.datasource”)**

@Data

@AllArgsConstructor

@Component

public class ConfigProprties

{

private String username;

private String password;

}

**How do you define custom bean scope?**

First we have to **implement this bean** on class which we will create custom bean scope.

Now we need to override bean methods.

**Can you explain when we should use singleton scope and prototype scope?**

Singleton

Database Configuration, Service layer, Application Configuration

Prototype

User Session, Thread Safe, Heavy Initialization

**What is difference between spring singleton and plain singleton?**

Spring Singleton scope within the application context and plain singleton scope within the JVM.

**What is the purpose of the BeanPostProcessor interface in Spring, and how can you use it to customize bean initialization and destruction?**

**What is the use of Profile in Spring Boot?**

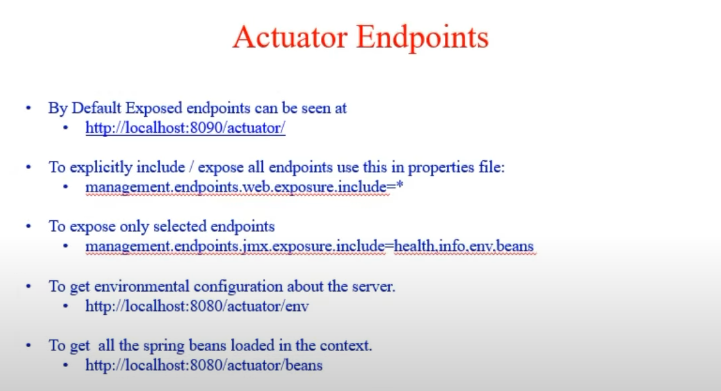
Dev environment

Prod environment

UAT or QA environment

**What is Spring Actuator? What are Advantages?**

* Actuator is a manufacturing term actually that is use for referring to mechanical devices to moving or controlling something.
* In Spring Boot, Actuator is addition feature that help us control manage and monitor our application spatially when its push to production.
* It include Auditing, Health, and matrices gathering means how many hit we got in HTTP and many more.
* We can enable this feature by adding the dependency “**spring-boor-starter-actuator**”.



What all Spring Boot starters you have used or what all modules you have worked on?

How will you run Spring Boot application?

What is the purpose of the @SpringBootApplication annotation in spring boot application?

Can I directly use above 3 annotation in my main class, instead of using @SpringBootApplication annotation, if yes will my application work as expected

What is AutoConfiguration in Spring Boot?

How can disable a specific auto configuration class in Spring Boot?

How can you customize the default configuration in Spring Boot?

How Spring Boot run() method works internally?

What is CommandLineRunner in Spring Boot?

