1 Spring IOC

1. IOC: IOC (Inversion of Control) is a process of managing objects' dependencies, the container will automatically inject the dependencies when the beans are created. IOC container in spring will manage the complete life cycle of them.
2. DI: Dependency injection is a design pattern, which serve the required resources (such as objects) an object depends on.
3. Dependency Injection Approaches
   - Constructor based: supports unit tests and immutable objects well.
   - Setter based: solves circular dependency problem, allows costly resources to be created as late as possible.
   - Field based: Syntax is concise and clear.
4. Bean Scope: refers to the lifecycle of beans, when the objects of beans will be created, how long will it live, how many objects will be instantiated.
5. Bean Lifecycle: refers to when and how beans are instantiated, what will they do while it lives, when and how they are destroyed.

2 Spring AOP

1. Spring AOP: Aspect oriented programming, which enables modularization of concerns, such as transaction management that cut across multiple types and objects.
2. Aspect: a modularization of a concern that cuts across multiple classes.
3. Advice: action taken by an aspect at a particular join point. Different types of advice include "around," "before" and "after" advice.
4. JoinPoint: a point during the execution of a program, such as the execution of a method or the handling of an exception.
5. PointCut: a predicate that matches join points. Advice is associated with a pointcut expression and runs at any join point matched by the pointcut
6. Target: objects being advised by one or more aspects.
7. @Transactional: A @Transactional annotation on a class specifies the default transaction semantics for the execution of any public operation in the class. A @Transactional annotation on a method within the class overrides the default transaction semantics given by the class annotation

3 Spring MVC

1. MVC: model is the application's dynamic data structure, view is a representation of information such as a chart, controller is the part which accepts input and converts it to commands for the model or view.
2. Spring MVC workflow: Request - Dispatcher Servlet - Controller - Model and View - View Resolver

4 Spring Boot

1. Spring boot advantages:
   - The way of configing database and java beans is flexible and easy.
   - Including embedded servlet container and works well with several servlet containers
   - Provides powerful batch processing and manages rest endpoints
   - Manages dependency well.
2. Spring boot Starter: handles the dependency management.
3. Auto Configuration: would automatically configure the Spring application based on the jar dependencies that the developer have added.
4. Rest API Design: CRUD operations:
   - create: post
   - read: get
   - update: put
   - delete: delete
5. Spring Restful API:
   - @RestController, @GetMapping, @PutMapping, @PostMapping, @DeleteMapping
   - @RequestParam, @PathVariable, @RequestBody, @ResponseBody
   - @Controller, @Service, @Repository

5 Exception Handling Process

1. @ExceptionHandler annotation is used to detect certain runtime exceptions and send responses according to the exception.
2. To make it available for all the classes in our project we just have to add the annotation @ControllerAdvice
3. @ResponseStatus annotation set the status code of an http request

6 Validation

1. @NotNull It determines that the value can't be null.
2. @Min It determines that the number must be equal or greater than the specified value.
3. @Max It determines that the number must be equal or less than the specified value.
4. @Size It determines that the size must be equal to the specified value.
5. @Pattern It determines that the sequence follows the specified regular expression.

7 Swagger

Swagger helps users build, document, test and consume RESTful web services.