How to handle exception

3 ways:

1. inside the Controller, use the annotation **@ExceptionHandler**
2. have a class implement **HandlerExceptionResolver** interface to become global
3. have a global controller class with **@ControllerAdvice** above and **@ExceptionHandler** inside.

AOP

Spring AOP is implemented based on **dynamic proxy** to provide the isolation of the business logic, improve the modularity of code and achieve the decoupling.

If the targetObject to be proxied implements an interface, it will use the **JDK proxy** to create the proxy class implementing the same interface(override the methods). based on **reflection** mechanism

If targetObject not implement any interface, it will use the **cglib** to create a subclass of targetObject to be the proxy class(override the methods of father class)

AOP annotation

**@Aspect** declares the class as aspect

**@Pointcut** declares the pointcut expression

**@Before** declares the before advice. It is applied before calling the actual method.

**@After** declares the after advice. It is applied after calling the actual method and before returning result.

**@AfterReturning** declares the after returning advice. It is applied after calling the actual method and before returning the result. But you can get the result value in the advice.

**@Around** declares the around advice. It is applied before and after calling the actual method.

**@AfterThrowing** declares the throws advice. It is applied if actual method throws exception.

spring boot and mongoDB experience

dependency:

**<artifactId>spring-boot-starter-data-mongodb</artifactId>**

application.yml:

| spring:  data:  mongodb:  host: 127.0.0.1  port: 27017  database: test  username: admin  password: 123456 |
| --- |

application.properties：

| spring.data.mongodb.host=127.0.0.1 spring.data.mongodb.port=27017 spring.data.mongodb.username=admin spring.data.mongodb.password=password spring.data.mongodb.database=DBTest spring.data.mongodb.authentication-database=admin |
| --- |

in Model: use @Document(collection = "XXXXXX") to connect the pojo with collections

in Controller: use@RestController above and use

@Autowired

**MongoTemplate mongoTemplate;**

and use @RequestMapping(“/insert”) @RequestMapping(“/update”) or remove

query language in Spring:

Query query = Query.query(Criteria.where("name").is("hello").and("password").is("world1"));

use mongoTemplate.find(query, XXXXXX.class)

.remove(query, XXXXXX.class)

.updateFirst(query,update,XXXXXX.class)

(Update update = new update)

config server

<artifactId>spring-cloud-config-server</artifactId>

use **@EnableConfigServer** at the main @SpringBootApplication to config server

how to load property

1. in spring.xml: <context:property-placeholder…>

in java file: @Value

1. in spring.xml: <util:properties…>

in java file: @Value

1. in java class: @PropertySource(value={"classpath:redis-key.properties"})

public class ReadProperties {

@Value(value="${jdbc.username}")

private String USER\_NAME;

how to manage the password, this kind of sensitive data

<artifactId>jasypt-spring-boot-starter</artifactId>

use the **JasyptStatelessService** interface and the .encrypt() method to encrypt the password

also use **BCryptPasswordEncoder**

(just provide the bean inside our passwordEncoder method)

@Bean

PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder(10);

}

how to deploy app

$ jar -xf myapp.jar

$ java org.springframework.boot.loader.JarLauncher

how to get the value in property file

@PropertySource

spring actuator

<artifactId>spring-boot-starter-actuator</artifactId>

spring actuator is used to monitor the spring boot application for things like health check, auditing, metrics collection, http tracking etc..

It has many endpoint like /health /metrics /loggers /info to track with URL to get diff info

how to connect to database in spring project

hibernate integrated SessionFactory

JPA integrated EntityManagerFactory

Also MongoTemplate to connect with MongoDB

how to write query to retrieve data from mongodb

db.collection.find().pretty()

also

insert: db.col.insert() (insertOne(), insertMany())

delete: db.col.remove()

update: db.col.update()

how to security your application

<artifactId>spring-boot-starter-security</artifactId>

application.xml:

| security:  basic:  enabled: true  user:  name: sunbufu  password: 123456 |
| --- |

JWT

-header

-payload

-signature

kafka experience and how do you use in your project

Topic: Every message published to a Kafka cluster has a category called a topic

Partition: Partition is a physical concept, and each Topic contains one or more Partitions

Usage:

1. Messaging
2. Website activity tracking
3. Metrics
4. Log Aggregation

what is the offset in kafka

offset is the index of every message to represent the position inside the partition.

LEO(Log End Offset) is the latest one in the queue.

replica and partition in kafka

Replica in kafka: a commit log to store in different broker

Partition: Partition is a physical concept, and each Topic contains one or more Partitions

Coding: **RestFul API**

@RestController

@RequestMapping("api/v1/person")

public class PersonController {

@Autowired

private PersonService personService;

@GetMapping

public List<Person> getAllPeople(){

return personService.getAllPeople();

}

@PostMapping

public void addPerson(Person person){

personService.addPerson();

}

@GetMapping(**path = "{id}"**)

public Person getPerson(**@PathVariable(“id”)** UUID id ){

return personService.getPersonById(id);

}

}

Coding: combine rest with http code

@GetMapping("/userinfo")

public ResponseEntity<Person> person() {

Person person = new Person();

person.setName("Ted");

return ResponseEntity.ok(person);

}

Coding: java stream List to Map

Map<Integer, Apple> appleMap = appleList.stream().collect(Collectors.toMap(Apple::getId, a -> a ,(k1,k2)->k1));

how to build docker image

1. get one used image to override
2. use Dockerfile to create one

what is EntryPoint

The purpose of ENTRYPOINT is the same as CMD, to specify the container launcher and its parameters, but ENTRYPOINT can also be substituted at runtime, although it is slightly more cumbersome than CMD, and needs to be specified with the --entrypoint argument to docker run.

Experience in k8s

kubernetes is a platform service to manage the containers’ workload and service.

k8s service:

1. load balancing
2. service discovery
3. automatically deployment and rollback
4. self-healing
5. key and configuration management

AWS experience

S3：file storage. In Java project we can use AmazonS3ClientBuilder api to manage the S3 repo

SNS: use for Email or TextMessage auto sending

SQS: Message queue between diff services

RDB: online database

EC2: cloud computer system

load balancer experience

using **AWS CLI** to create load balancer to automatically distribute incoming traffic among multiple targets, such as EC2 instances, containers.

how to control the traffic in AWS

**Amazon route 53**: route 53 can monitor the running status of the resource, also we can create an Amazon **CloudWatch** alert ourselves to get Notification when service has issue or down.

Kafka vs SQS

Kafka:

1. pretty **scalable** system and fits on **high workloads**
2. have a good message **throughput**
3. Kafka topic consists of some number of **partitions** which can be read completely **parallel by different consumers** in one **consumer group**
4. good performance

SQS:

1. SQS is an **managed** service (do not have to support infrastructure by yourself)
2. SQS is better for eventing when you need to catch some message, it will **automatically popped out** from the queue

MQ(Message Queue) Advantage

1. decoupling
2. peak shaving
3. asynchronous

What happen if one of consumer shutdown in Kafka

If the consumer crashes or is shut down, its partitions will be re-assigned to another member, which will begin consumption from the last committed offset of each partition. If the consumer crashes before any offset has been committed, then the consumer which takes over its partitions will use the reset policy.

What happen if consumer added back to consumer group

Rebalancing

Jenkins pipeline

config cmd in **Jenkinsfile** based on Docker to run

What is context switching

In computer programming, context switching refers to **storing** an active process in its **current state** so the system's CPU can shift its resources to other tasks. A user can then instruct the CPU at any time to resume the process from the same point.

How to debug in MicroService

1. Make Sure Logs are Searchable.
2. Return Transactional References Back to the Client.
3. Invest in Setting Up a Logging Framework.
4. Consider Monitoring Tools.

What is API gateway

An API gateway is an API management tool that sits between **a client and a collection of backend services**.

All requests from clients **first** go through the API Gateway. It then routes requests to the appropriate microservice. The API Gateway will often handle a request by invoking multiple microservices and aggregating the results

How do you design table with many to many relation

To establish a many-to-many relationship between two or more tables, the simplest way is to use a **Junction Table**

Override one of **equals** or **hashcode**, what will happen

You **must override hashCode() in every class that overrides equals()**. Failure to do so will result in a violation of the general contract for Object.hashCode(), which will prevent your class from functioning properly in conjunction with all **hash-based** collections, including HashMap, HashSet, and Hashtable.

When will hashmap give bad performance

it slows down due to intense GC(garbage collection) and then it breaks with OutOfMemoryError

what is transient and give a real world example

transient in java means can not be serialized

GC algorithms

**mark-and-sweep** algorithm

Atomic integer

(using **volatile** & CAS (compare and swap))

The whole reason for atomic variables is for synchronization purposes during multi-threading operations.