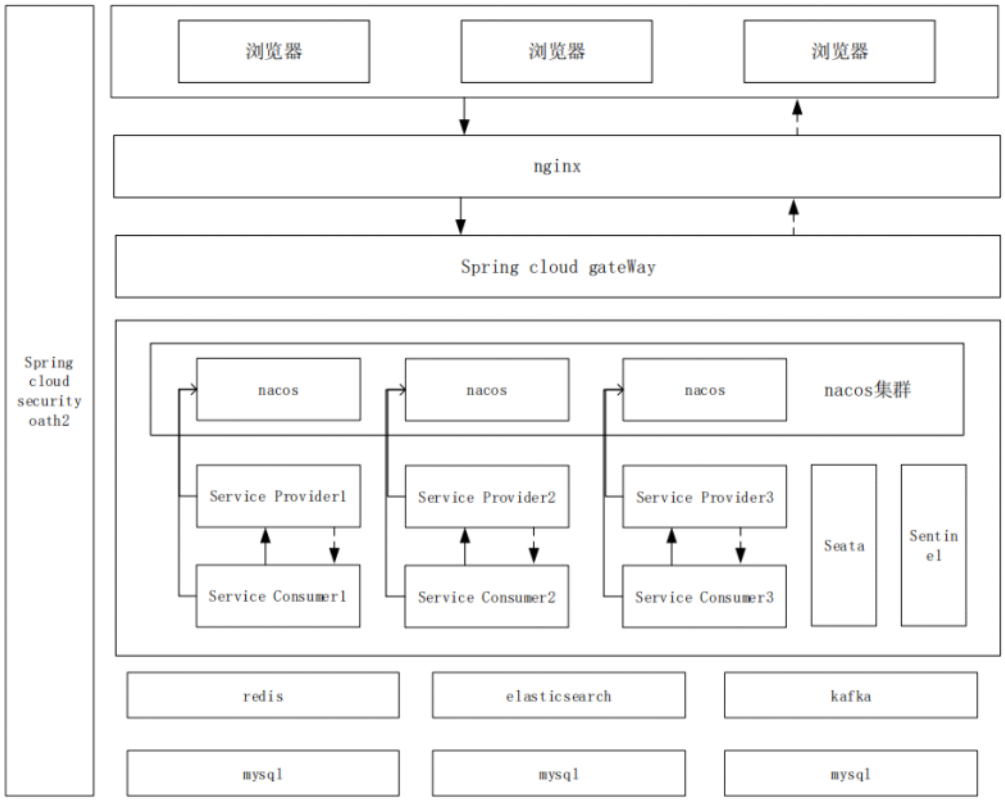
基础工程应用平台建设

2020年5月28日 9:52

模块设计规划



- 采用spring cloud alibaba作为整体微架构
- 初期规划模块

主工程——单一Web服务

web	SpringCloud接口暴露层-提供给前端访问层
web-service	服务提供层（初期可不考虑web层，直接以web-service返回）
common	公共配置、全局统一异常、全局响应、以及工具类支持模块
generator	代码生成模块
config	基于nacos的配置中心（初期工程以及代码请忽略）
gateway	微服务网关
task	定时任务模块
seata	seata-account-service、seata-order-service、seata-storage-service 分布式事务案例（初期工程以及代码请忽略）
swagger	各服务引入本模块即可支持swagger扫描

初期接入服务设计

nacos	各服务注册中心、配置中心，可利用Nginx做集群
spring cloud Security（Oath2）	认证授权服务（认证、授权可拆分）
spring Cloud Gateway	服务网关
file Upload	文件上传服务（各服务统一）
feign Service	远程服务
swagger service	接口文档在线可视化、调试

初期接入中间件设计

nginx	反向代理、指向前端独立应用
redis	全局缓存、oauth2认证存储容器
elasticsearch	搜索引擎
mysql	数据存储中心

本次架构设计/规范定义基本原则
以过往遇到的某个项目为例

原生的SQL，开发效率慢，可引入ORM思想优化，实际落地采用MybatisPlus或者SpringData系列 需要考虑分页统一问题 尽量使用单一持久层

部分存在性能问题的代码 考虑缓存 例如 持久化用户角色权限

数据库表中的主要时间字段ctime utime，部分表存在空缺 考虑jdk8中的时间类

数据库表主键 采用 全局唯一雪花算法生成的ID 框架提供 uuid性能较差 官方不推荐

数据的响应、异常，目前代码混乱，没有一个统一的风格，存在大量硬编码

以包分类区分模块，维护性很一般 考虑 拆分模块 构建单工程 多模块应用

公用代码、工具类 需要统一，项目中大量使用 规范不一 避免自行创建 首先去common模块查找 所有公共文件类 移动到common模块 如果业务强依赖的工具类 则放置到自己的模块使用

注释规范

- 所有的类 必须有文件头注释 至少具备 （作者、描述）
- 复杂业务建议带上一定注释
- service接口需要有注释、包括入参 返回结果
- 类名首字母大写
- bean对象 需要具备swagger属性注释 以及属性注释

Code码

基础Code码全局统一，通过枚举类定义 例如

SUCCESS(0, "请求成功"),
SYSTEM_BUSY(100, "系统繁忙"),
REQUEST_TIME_OUT(300, "请求超时"),
PARAMETER_ERROR(400, "参数错误"),
NETWORK_ERROR(404, "网络异常"),
DATA_NOT_EXISTS(600, "数据不存在"),
ACCESSDENIED_ERROR(501,"你无权访问"),
FAILURE(999, "未知错误");

各业务Code码，自行定义 通过约定 不允许占用同一Code段
PARAM_INVALID(3001, "菜单不存在"),
MENU_ROLE_ERROR(3002, "菜单、权限对应数据有误"),
ISEXIT_CHILDREN(3003, "当前节点下存在子菜单,禁止删除"),
PARENT_NODE_QUERY_NULL(3004, "父节点查询不到数据、权限对应数据有误"),

PARENT_IS_NOTEXIT(3005, "父菜单不存在"),
ROLE_ONEMENU_ISNULL(3006, "当前用户角色无一级菜单,不允许添加子菜单");

返回数据格式（用于前端对接）：

1.普通查询 GET

{"code":0,"msg":"请求成功","data":{"cuser":"black","ctime":null,"uuser":"black","utime":null,"id":"1","username":"admin","password":"$2a$10
$ZsM5tnQRh0GR5CSg.Kdm.eXGVHcyoFwQB9rfgKSEEntAeo6A67yjO"}}

2.提交POST

{"code":0,"msg":"请求成功","data": "true"}

3.修改PUT

{"code":0,"msg":"请求成功","data": "true"}

4.分页查询　案例 ?pageNum=1&pageSize=10&id=1　　pageNum=起始页数　pageSize=页数量

{"code":0,"msg":"请求成功","data":{"records":[{"cuser":"black","ctime":null,"uuser":"black","utime":null,"id":"1195225356584357890","username":" black","password":"$2a$10
$ZsM5tnQRh0GR5CSg.Kdm.eXGVHcyoFwQB9rfgKSEEntAeo6A67yjO"},{"cuser":"black","ctime":null,"uuser":"black","utime":null,"id":"119 5225358736035842","username":"black",
"password":"$2a$10
$ZsM5tnQRh0GR5CSg.Kdm.eXGVHcyoFwQB9rfgKSEEntAeo6A67yjO"},{"cuser":"black","ctime":null,"uuser":"black","utime":null,"id":"119 5225362192142338","username":"black",
"password":"$2a$10$ZsM5tnQRh0GR5CSg.Kdm.eXGVHcyoFwQB9rfgKSEEntAeo6A67yjO"}],"total":6,"size":10,"current":1,"orders":[],"sear chCount":true,"pages":1}}

返回结果

code	成功状态码
msg	前台返回提示消息
data	数据整体
records	仅数据
total	数据总数
size	当前页数量
current	当前页
pages	共计页数

LOG日志——logBack

不过多规范，每个访问方法 需要有 关键业务代码 需要有 Log记录　方便调用定位、问题定位、追踪

备注：

若采用mybatis plus等框架无法解决的业务 则手动创建xml配置 手写sql 分页则使用pagehelper等插件 最终统一分页的数据格式返回

微服务规范设计篇

2020年6月1日 17:17

parent作为父工程

pom中 定义依赖管理器

<dependencyManagement>
…….
</dependencyManagement>

每个微服务都需要具备独立的名称以及访问前缀

网关的路由ID、触发条件 跟微服务名称尽量保持一致

例子

```
gateway:
  routes:
    - id: constomer-server #路由ID，没有固定规则但要求唯一，建议配合服务名
      #uri: http://localhost:9100      #匹配后提供服务的IP地址 不推荐
      uri: lb://constomer-server #匹配后提供服务的路由地址
      predicates:
        - Path=/constomer-server/**          # 断言，路径相匹配的进行路由
      filters:
      #- StripPrefix=1 #跳过前缀 序号为第几个/
      #- PrefixPath=/auth 添加前缀
```

Nacos初始化本地配置

2020年6月1日 17:18

http://nacos.io/

本次使用的版本是来自 1.2.0
https://github.com/alibaba/nacos/releases

默认的nacos是基于文件的存储 我们希望能够把相关的配置存储到mysql数据库中 以保证不容易丢失 和方便管理

那么具体的做法（基本不用再做配置文件调整，因为已经调整好了）

关键配置文件的熟悉开启 调整为你自己的数据库 nacos\conf\application.properties

application.properties

```
#*************** Spring Boot Related Configurations ***************#
### Default web context path:
server.servlet.contextPath=/nacos
### Default web server port:
server.port=8848

#*************** Network Related Configurations ***************#
### If prefer hostname over ip for Nacos server addresses in cluster.conf:
# nacos.inetutils.prefer-hostname-over-ip=false

### Specify local server's IP:
# nacos.inetutils.ip-address=



#*************** Config Module Related Configurations ***************#
### If user MySQL as datasource:
spring.datasource.platform=mysql

### Count of DB:
db.num=1

## Connect URL of DB:
db.url.0=jdbc:mysql://127.0.0.1:3306/black?characterEncoding=utf8&connectTimeout=1000&socketTimeout=3000&autoReconnect=true
db.user=root
db.password=123456
```

然后在对应的数据库中 导入nacos官方提供的数据库脚本 nacos-mysql.sql

启动 windows下
\nacos\bin\startup.cmd
如果手动点击黑屏 黑窗一闪而过 在当前目录下 dos窗口 执行

```
C:\Users\Administrator\Desktop\基础架构3.0\nacos\bin>startup.cmd
Please set the JAVA_HOME variable in your environment, We need java(x64)! jdk8 or later is better!
```

这里是要配置jdk 具体参照网上做法 java-

参考文章https://jingyan.baidu.com/article/0202781175839b1bcc9ce529.html 然后再次启动即可

启动完成的样子，尽量还是避免有中文路径 不推荐像图中这样

http://localhost:8848/nacos/#/login 登录路径 必须全部给上

账号密码均为nacos 如果8.0的数据库需要继续往下看

8.0的mysql数据库 需要到nacos目录下创建

下载8.0的jar 依赖

1.将下载好的jar 传到nacos的根目录下新建plugins/mysql文件夹下

2.nacos配置文件中 数据库路径调整一下　不然会报编码集、时间方面的支持问题
jdbc:mysql://localhost:3306/black?useUnicode=true&characterEncoding=UTF-8
&useSSL=false&serverTimezone=Asia/Shanghai&zeroDateTimeBehavior=CONVERT_TO_
NULL

https://github.com/alibaba/nacos/pull/855 该问题提出与解决

Nacos Config配置中心

2020年6月1日 17:18

需要application定义启动环境
bootstrap定义规则

如果想要获取到nacos上的配置信息 nacos上的文件格式定义为
spring.applcation.name-active.file-extension

服务名称-环境.文件后缀
具体表现为 config-client-dev.yml

在nocas上面需要创建这样的文件 然后填写信息 即可被读取

group可以自己定义 用来做区分 namespace 只能填写id

Sentinel服务容错

2020年6月1日 17:19

版本sentinel-dashboard-1.7.0

sentinel的应用需要注意的主要是 持久化配置 每次服务重启 默认是会丢失配置的

基于nacos去做

要引用这个依赖 然后配置上正确的dataId group等信息

```
 <dependency>
        <groupId>com.alibaba.csp</groupId>
        <artifactId>sentinel-datasource-nacos</artifactId>
</dependency>
```

yml配置

```
  sentinel:
    transport:
      #可视化连接
      dashboard: 127.0.0.1:8080
      #通信端口
      port: 8719
    datasource:
      ds1:
        nacos:
          server-addr: 127.0.0.1:8848
          #namespace: c22e5019-0bee-43b1-b80b-fc0b9d847501
          dataId: ${spring.application.name}
          group: DEFAULT_GROUP
          data-type: json
          rule-type: flow
```

需要建立一样的dataId group

在nacos上添加配置文件 填写json配置 不再需要文件后缀名

```
[
    {
        "resource": "/hello",
        "limitApp": "default",
        "grade": 1,
        "count": 5,
        "strategy": 0,
        "controlBehavior": 0,
        "clusterMode": false
    }
]
```

Seata分布式事务

2020年6月1日 17:20

版本0.9.0
1.0.0经过测试 似乎问题颇多

需要创建一个seata库 导入官方sql db_store.sql

需要在各业务库中 导入db_undo_log.sql

然后修改file.conf registry.conf 两个配置文件

然后项目的各子服务 加入依赖的同时 各resources下 也要有file.conf registry.conf 两个配置文件

```yaml
spring:
  application:
    name: seata-storage-service
  cloud:
    alibaba:
      seata:
        tx-service-group: yaoshun_tx_group
    nacos:
      discovery:
        server-addr: localhost:8848
```

yml配置

```
service {

  vgroup_mapping.my_test_tx_group = "yaoshun_tx_group" #修改自定义事务组名称

  default.grouplist = "127.0.0.1:8091"
  enableDegrade = false
  disable = false
  max.commit.retry.timeout = "-1"
  max.rollback.retry.timeout = "-1"
  disableGlobalTransaction = false
}
```

seata服务file配置

```
service {
  vgroup_mapping.yaoshun_tx_group = "default" #修改自定义事务组名称

  default.grouplist = "127.0.0.1:8091"
  enableDegrade = false
  disable = false
  max.commit.retry.timeout = "-1"
  max.rollback.retry.timeout = "-1"
  disableGlobalTransaction = false
}
```

java工程服务中file配置

mysql持久化 各配置都基本一样

```
55    ## store mode: file、db
56    mode = "db"
57
58    ## file store
59    file {
60      dir = "sessionStore"
61
62      # branch session size , if exceeded first try compress lockkey, still exceeded throws exceptions
63      max-branch-session-size = 16384
64      # globe session size , if exceeded throws exceptions
65      max-global-session-size = 512
66      # file buffer size , if exceeded allocate new buffer
67      file-write-buffer-cache-size = 16384
68      # when recover batch read size
69      session.reload.read_size = 100
70      # async, sync
71      flush-disk-mode = async
72    }
73
74    ## database store
75    db {
76      ## the implement of javax.sql.DataSource, such as DruidDataSource(druid)/BasicDataSource(dbcp) etc.
77      datasource = "dbcp"
78      ## mysql/oracle/h2/oceanbase etc.
79      db-type = "mysql"
80      driver-class-name = "com.mysql.jdbc.Driver"
81      url = "jdbc:mysql://127.0.0.1:3306/seata"
82      user = "root"
83      password = "123456"
84      min-conn = 1
85      max-conn = 3
86      global.table = "global_table"
87      branch.table = "branch_table"
88      lock-table = "lock_table"
89      query-limit = 100
90    }
```