

第二章 新闻列表

1. 新闻列表

1.1 需求

学堂资讯

更多

题库

升学

其他

初中历史易错易混淆的知识点（中国古代史）

中国历史年表记忆口诀

励志题库知识点-道德与法治

中考历史知识点：古老的亚洲文明

中考道法热点：增强文化自信

网站首页会有相关的新闻资讯显示，点进去会有详情显示

首页 > 学堂资讯 > 资讯详情

初中历史易错易混淆的知识点（中国古代史）

时间：2020年11月20日 14:08:39 编辑: >

点击更多，会进入到列表详情：

- 题库资讯
- 升学资讯
- 其他资讯



初中历史易错易混淆的知识点（中国古代史）

初中历史易错易混淆的知识点（中国古代史）

2020年11月20日 14:08:39

中国历史年表记忆口诀



中国历史年表记忆口诀

2020年11月20日 13:48:41

1.2 数据库设计

<div> <div>保存</div> <div>添加字段</div> <div>插入字段</div> <div>删除字段</div> <div>主键</div> <div>上移</div> <div>下移</div> </div>									
字段	索引	外键	触发器	选项	注释	SQL 预览			
名							类型	长度	小数点
id							bigint	0	0
title							varchar	255	0
summary							varchar	255	0
image_url							varchar	255	0
content							text	0	0
tab							tinyint	0	0
create_time							bigint	0	0
author							varchar	255	0
n_status							tinyint	0	0

1.3 步骤分析

1. 首页列表实现，每一个标签只显示5条最新的数据，查询条件中根据标签查询，每次切换标签都访问一次接口
2. 列表查询sql语句，只查询题目以及id，提高查询性能
3. 查询出结果后，放入redis缓存中，设置2分钟的过期时间，加快访问速度的同时也可以防止出现集中访问的情况，应对短暂的热点现象
4. 文章详情，根据id查询文章的详细信息，同样放入缓存中，考虑到文章内容过大，设计30秒的过期时间,后期如果出现访问量激增的情况，考虑放入es中
5. 列表详情页面，分页查询，同样放入缓存，sql语句只查询需要的信息即可
6. 可以写缓存的通用实现

1.4 编码

1.4.1 创建web模块

1. pom文件 设置package 为pom
2. 新建mszlu-xt-web-api, mszlu-xt-web-model, mszlu-xt-web-domain, mszlu-xt-web-dao, mszlu-xt-web-service, mszlu-xt-web-service-impl模块
3. 依赖包和sso模块一致
4. 配置文件也和sso一致, 数据库实际业务中分为多个, 在这里统一使用mszlu-xt数据库
5. 建模块是大家在项目阶段要学习的一个重要实践知识点, 必须亲手能搭建出来

1.4.2 编码

1. 创建NewsController, 写三个接口

```
package com.mszlu.xt.web.api;

import com.mszlu.common.model.CallResult;
import com.mszlu.xt.model.service.NewsService;
import com.mszlu.xt.web.model.params.NewsParam;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping("news")
public class NewsApi {

    @Autowired
    private NewsService newsService;

    @PostMapping("newsList")
    public CallResult newsList(@RequestBody NewsParam newsParam){
        return newsService.newsList(newsParam);
    }

    @PostMapping("detail")
    public CallResult news(@RequestBody NewsParam newsParam){
        return newsService.findNewsById(newsParam);
    }
}
```

```

    }

    @PostMapping("newsDetailList")
    public CallResult newsDetailList(@RequestBody NewsParam
newsParam){
        return newsService.newsDetailList(newsParam);
    }
}

```

参数设置:

```

package com.mszlu.xt.web.model.params;

import lombok.Data;

@Data
public class NewsParam {
    private int page = 1;
    private int pageSize = 20;
    private Long id;
    private Integer tab;

    private String title;
    private String summary;
    private String imageUrl;
    private String content;
    private Long createTime;
    private String author;
    /**
     * 0 正常 1 删除
     */
    private Integer status;
}

```

2. NewsService

```

package com.xiaopizhu.web.service;

import com.xiaopizhu.common.model.CallResult;
import com.xiaopizhu.web.model.params.NewsParam;

```

```
public interface NewsService {

    CallResult newsList(NewsParam newsParam);

    CallResult newsDetailList(NewsParam newsParam);

    CallResult findNewsById(NewsParam newsParam);

}
```

```
package com.xiaopizhu.web.service.impl;

import com.xiaopizhu.common.model.CallResult;
import com.xiaopizhu.common.service.AbstractTemplateAction;
import com.xiaopizhu.web.domain.NewsDomain;
import com.xiaopizhu.web.domain.repository.NewsDomainRepository;
import com.xiaopizhu.web.model.params.NewsParam;
import com.xiaopizhu.web.service.NewsService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class NewsServiceImpl extends AbstractService implements
NewsService {

    @Autowired
    private NewsDomainRepository newsDomainRepository;

    @Override
    public CallResult newsList(NewsParam newsParam) {
        NewsDomain newsDomain =
this.newsDomainRepository.createDomain(newsParam);
        return this.serviceTemplate.exeOnSlave(new
AbstractTemplateAction<Object>() {
            @Override
            public CallResult<Object> checkParam() {
                return newsDomain.checkNewsListParam();
            }

            @Override
            public CallResult<Object> doAction() {
                return newsDomain.newsList(false);
            }
        });
    }
}
```

```

        }
    });
}

@Override
public CallResult findNewsById(NewsParam newsParam) {
    NewsDomain newsDomain =
this.newsDomainRepository.createDomain(newsParam);
    return this.serviceTemplate.exeOnSlave(new
AbstractTemplateAction<Object>() {
        @Override
        public CallResult<Object> doAction() {
            return newsDomain.findNewsById();
        }
    });
}

@Override
public CallResult newsDetailList(NewsParam newsParam) {
    NewsDomain newsDomain =
this.newsDomainRepository.createDomain(newsParam);
    return this.serviceTemplate.exeOnSlave(new
AbstractTemplateAction<Object>() {
        @Override
        public CallResult<Object> doAction() {
            return newsDomain.newsList(true);
        }
    });
}
}
}

```

3. domain实现具体的业务逻辑

```

package com.mszlu.xt.web.domain;

import com.baomidou.mybatisplus.extension.plugins.pagination.Page;
import com.mszlu.common.model.BusinessCodeEnum;
import com.mszlu.common.model.CallResult;
import com.mszlu.common.model.ListPageModel;
import com.mszlu.xt.pojo.News;
import com.mszlu.xt.web.domain.repository.NewsDomainRepository;

```

```
import com.mszlu.xt.web.model.NewsModel;
import com.mszlu.xt.web.model.params.NewsParam;
import org.joda.time.DateTime;
import org.springframework.beans.BeanUtils;

import java.util.ArrayList;
import java.util.List;

public class NewsDomain {

    private NewsDomainRepository newsDomainRepository;
    private NewsParam newsParam;

    public NewsDomain(NewsDomainRepository newsDomainRepository,
NewsParam newsParam) {
        this.newsDomainRepository = newsDomainRepository;
        this.newsParam = newsParam;
    }

    public NewsModel copy(News news){
        if (news == null){
            return null;
        }
        NewsModel newsModel = new NewsModel();
        //属性copy
        BeanUtils.copyProperties(news,newsModel);
        if (news.getCreateTime() != null) {
            newsModel.setCreateTime(new
DateTime(news.getCreateTime()).toString("yyyy年MM月dd日 HH:mm:ss"));
        }
        if (news.getImageUrl() != null) {
            if (!news.getImageUrl().startsWith("http")) {

                newsModel.setImageUrl(newsDomainRepository.qiniuConfig.getFileServ
erUrl() + news.getImageUrl());
            }
        }
        return newsModel;
    }

    public List<NewsModel> copyList(List<News> newsList){
        List<NewsModel> newsModelList = new ArrayList<>();
        for (News news : newsList){
            newsModelList.add(copy(news));
        }
    }
}
```

```

    }
    return newsModelList;
}

public CallResult<Object> checkNewsListParam() {
    if (newsParam.getPageSize() > 5){
        newsParam.setPageSize(5);
    }
    if (newsParam.getTab() == null){
        return
        CallResult.fail(BusinessCodeEnum.CHECK_PARAM_NO_RESULT.getCode(),"p
aram error: tab is null");
    }
    return CallResult.success();
}

public CallResult<Object> newsList(boolean isDetail) {
    int page = this.newsParam.getPage();
    int pageSize = this.newsParam.getPageSize();
    Integer tab = this.newsParam.getTab();
    Page<News> newsPage =
    this.newsDomainRepository.findNewsListByTab(page,pageSize,tab,isDet
ail);
    ListPageModel<NewsModel> listPageModel = new
    ListPageModel<>();
    List<News> result = newsPage.getRecords();
    List<NewsModel> newsModelList = copyList(result);
    listPageModel.setList(newsModelList);
    listPageModel.setPage(page);
    listPageModel.setPageSize(pageSize);
    listPageModel.setPageCount(newsPage.getPages());
    listPageModel.setSize(newsPage.getTotal());
    return CallResult.success(listPageModel);
}

public CallResult<Object> findNewsById() {
    Long id = this.newsParam.getId();
    News news =
    this.newsDomainRepository.findNewsDetailById(id);
    NewsModel newsModel = copy(news);
    return CallResult.success(newsModel);
}
}

```



```
package com.mszlu.xt.web.domain.repository;

import
com.baomidou.mybatisplus.core.conditions.query.LambdaQueryWrapper;
import com.baomidou.mybatisplus.extension.plugins.pagination.Page;
import com.mszlu.xt.web.dao.NewsMapper;
import com.mszlu.xt.web.dao.data.News;
import com.mszlu.xt.web.domain.NewsDomain;
import com.mszlu.xt.web.domain.qiniu.QiniuConfig;
import com.mszlu.xt.web.model.enums.Status;
import com.mszlu.xt.web.model.params.NewsParam;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

import javax.annotation.Resource;

@Component
public class NewsDomainRepository {

    @Autowired
    public QiniuConfig qiniuConfig;

    @Resource
    private NewsMapper newsMapper;

    public NewsDomain createDomain(NewsParam newsParam) {
        return new NewsDomain(this, newsParam);
    }

    public Page<News> findNewsListByTab(int page, int pageSize,
Integer tab, boolean isDetail) {
        Page<News> newsPage = new Page<>(page, pageSize);
        LambdaQueryWrapper<News> queryWrapper = new
LambdaQueryWrapper<>();
        queryWrapper
            .eq(News::getTab, tab)
            .eq(News::getStatus, Status.NORMAL.getCode());
        if (isDetail){

            queryWrapper.select(News::getId, News::getImageUrl, News::getCreateTime,
```

```

News::getAuthor,News::getTitle,News::getTab,News::getSummary);
    }else {
        queryWrapper.select(News::getId, News::getTitle);
    }
    return newsMapper.selectPage(newsPage,queryWrapper);
}

    public News findNewsById(Long id) {
        LambdaQueryWrapper<News> queryWrapper = new
LambdaQueryWrapper<>();

        queryWrapper.eq(News::getId,id).select(News::getId,News::getTitle)
;
        return newsMapper.selectOne(queryWrapper);
    }

    public News findNewsDetailById(Long id) {
        LambdaQueryWrapper<News> queryWrapper = new
LambdaQueryWrapper<>();
        queryWrapper.eq(News::getId,id);
        return newsMapper.selectOne(queryWrapper);
    }
}

```

```

package com.mszlu.xt.web.dao;

import com.baomidou.mybatisplus.core.mapper.BaseMapper;
import com.mszlu.xt.web.dao.data.News;

public interface NewsMapper extends BaseMapper<News> {
}

```

4. 其他代码，七牛云图片地址配置，枚举，分页模型等

```

package com.mszlu.common.model;

import lombok.Data;

import java.io.Serializable;

```

```

import java.util.ArrayList;
import java.util.List;

/**
 * @author Jarno
 */
@Data
public class ListPageModel<T> implements Serializable {

    private long pageCount;

    private int page;

    private int pageSize;

    private long size;

    private List<T> list;

    public ListPageModel<T> initNull(){
        ListPageModel<T> listModel = new ListPageModel<T>();
        listModel.setList(new ArrayList<T>());
        listModel.setPage(1);
        listModel.setPageCount(1);
        listModel.setSize(0);
        return listModel;
    }

}

```

```

package com.mszlu.xt.web.model.enums;

import java.util.HashMap;
import java.util.Map;

/**
 * @author Jarno
 */
public enum Status {

    /**
     * look name
     */
}

```

```
NORMAL(0,"正常"),  
DELETE(1,"删除");
```

```
private static final Map<Integer, Status> CODE_MAP = new  
HashMap<>(3);
```

```
static{  
    for(Status status: values()){  
        CODE_MAP.put(status.getCode(), status);  
    }  
}
```

```
/**  
 * 根据code获取枚举值  
 * @param code  
 * @return  
 */  
public static Status valueOfCode(int code){  
    return CODE_MAP.get(code);  
}
```

```
private int code;  
private String msg;
```

```
Status(int code, String msg) {  
    this.code = code;  
    this.msg = msg;  
}
```

```
public int getCode() {  
    return code;  
}
```

```
public void setCode(int code) {  
    this.code = code;  
}
```

```
public String getMsg() {  
    return msg;  
}
```

```
public void setMsg(String msg) {  
    this.msg = msg;  
}
```

```
}
```

```
package com.mszlu.xt.web.domain.qiniu;

import lombok.Data;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Configuration;

@Configuration
@Data
public class QiniuConfig {

    @Value("${qiniu.file.server.url}")
    private String fileServerUrl;
}
```

```
package com.mszlu.xt.web.dao.data;

import com.baomidou.mybatisplus.annotation.TableField;
import lombok.Data;

/**
 * 新闻
 */
@Data
public class News {
    private Long id;
    private String title;
    private String summary;
    private String imageUrl;
    private String content;
    /**
     * 1 题库 2 升学 3 其他
     */
    private Integer tab;
    private Long createTime;
    private String author;
    /**
     * 0 正常 1 删除
     */
    @TableField("n_status")
```

```
        private Integer status;  
    }  
}
```

```
package com.mszlu.xt.web.model;  
  
import lombok.Data;  
  
/**  
 * 新闻  
 */  
@Data  
public class NewsModel {  
    private Long id;  
    private String title;  
    private String summary;  
    private String imageUrl;  
    private String content;  
    private Integer tab;  
    private String createTime;  
    private String author;  
}
```

5. 测试

1.4.3 添加缓存支持

1. 添加依赖, common模块

```
<dependency>  
    <groupId>org.springframework.boot</groupId>  
    <artifactId>spring-boot-starter-data-redis</artifactId>  
</dependency>  
<dependency>  
    <groupId>org.springframework.boot</groupId>  
    <artifactId>spring-boot-starter-aop</artifactId>  
</dependency>
```

2. 添加redis配置

```
#redis 配置 本机开发使用的为单机模式
spring.redis.host=localhost
spring.redis.port=6379

# redis集群配置 生产环境使用集群配置
#spring.redis.jedis.pool.max-wait=5000ms
#spring.redis.jedis.pool.max-Idle=100
#spring.redis.jedis.pool.min-Idle=10
#spring.redis.timeout=10s
#spring.redis.cluster.nodes=192.168.111.131:6379,192.168.111.131:6380,192.168.111.131:6381
#spring.redis.cluster.max-redirects=5
```

3. 在common模块中，添加缓存支持，使用AOP实现

```
package com.mszlu.common.cache.redis;

import com.alibaba.fastjson.JSON;
import com.mszlu.common.cache.Cache;
import com.mszlu.common.model.CallResult;
import lombok.extern.slf4j.Slf4j;
import org.apache.commons.codec.digest.DigestUtils;
import org.apache.commons.lang3.StringUtils;
import org.aspectj.lang.ProceedingJoinPoint;
import org.aspectj.lang.Signature;
import org.aspectj.lang.annotation.Around;
import org.aspectj.lang.annotation.Aspect;
import org.aspectj.lang.annotation.Pointcut;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.redis.core.RedisTemplate;
import org.springframework.stereotype.Component;

import java.lang.reflect.Method;
import java.time.Duration;

@Aspect
@Component
@Slf4j
public class CacheAspect {

    @Autowired
    private RedisTemplate<String, String> redisTemplate;
```

```

@Pointcut("@annotation(com.msزلu.common.cache.Cache)")
public void pt(){}

@Around("pt()")
public Object around(ProceedingJoinPoint pjp){
    try {
        Signature signature = pjp.getSignature();
        //类名
        String className =
pjp.getTarget().getClass().getSimpleName();
        //调用的方法名
        String methodName = signature.getName();

        Class[] parameterTypes = new
Class[pjp.getArgs().length];
        Object[] args = pjp.getArgs();
        //参数
        String params = "";
        for(int i=0; i<args.length; i++) {
            if(args[i] != null) {
                params += JSON.toJSONString(args[i]);
                parameterTypes[i] = args[i].getClass();
            }else {
                parameterTypes[i] = null;
            }
        }
        if (StringUtils.isNotEmpty(params)) {
            //加密 以防出现key过长以及字符转义获取不到的情况
            params = DigestUtils.md5Hex(params);
        }
        Method method =
pjp.getSignature().getDeclaringType().getMethod(methodName,
parameterTypes);
        //获取Cache注解
        Cache annotation = method.getAnnotation(Cache.class);
        //缓存过期时间
        long expire = annotation.expire();
        //缓存名称
        String name = annotation.name();
        //先从redis获取
        String redisKey = name + "::" +
className+"::"+methodName+"::"+params;
    }
}

```



```

        String redisValue =
redisTemplate.opsForValue().get(redisKey);
        if (StringUtils.isNotEmpty(redisValue)){
            log.info("走了缓存~~~, {}, {}", className, methodName);
            return JSON.parseObject(redisValue,
CallResult.class);
        }
        Object proceed = pjp.proceed();

        redisTemplate.opsForValue().set(redisKey, JSON.toJSONString(proceed
), Duration.ofMillis(expire));
        log.info("存入缓存~~~ {}, {}", className, methodName);
        return proceed;
    } catch (Throwable throwable) {
        throwable.printStackTrace();
    }
    return CallResult.fail();
}
}

```

```

package com.mszlu.common.cache;

import java.lang.annotation.*;

@Target({ElementType.METHOD})
@Retention(RetentionPolicy.RUNTIME)
@Documented
public @interface Cache {

    long expire() default 1 * 60 * 1000;

    String name() default "";
}

```

4. 在Controller层 接口进行应用

```

package com.mszlu.xt.web.api;

```

```

import com.mszlu.common.cache.Cache;
import com.mszlu.common.model.CallResult;
import com.mszlu.xt.model.service.NewsService;
import com.mszlu.xt.web.model.params.NewsParam;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping("news")
public class NewsApi {

    @Autowired
    private NewsService newsService;

    @PostMapping("newsList")
    @Cache(expire = 2 * 60 * 1000, name = "News")
    public CallResult newsList(@RequestBody NewsParam newsParam){
        return newsService.newsList(newsParam);
    }

    @PostMapping("detail")
    @Cache(expire = 30 * 1000, name = "News")
    public CallResult news(@RequestBody NewsParam newsParam){
        return newsService.findNewsById(newsParam);
    }

    @PostMapping("newsDetailList")
    @Cache(expire = 30 * 1000, name = "News")
    public CallResult newsDetailList(@RequestBody NewsParam
newsParam){
        return newsService.newsDetailList(newsParam);
    }
}

```

5. 测试

```

<==      Columns: id, title
<==      Row: 1, 多地教育部宣布, 中小学寒假时间定了, 学生欲哭无泪但还有好消息
<==      Total: 1
Closing non transactional SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@1a5824f]
2021-04-05 00:17:09.699 INFO 158648 --- [nio-8118-exec-1] c.x.c.service.impl.ServiceTemplateImpl : slave datasource method run time:260ms
2021-04-05 00:17:09.707 INFO 158648 --- [nio-8118-exec-1] c.x.c.cache.redis.config.CacheAspect : 存入缓存~~~ NewsApi,newsList
2021-04-05 00:17:20.103 INFO 158648 --- [nio-8118-exec-2] c.x.c.cache.redis.config.CacheAspect : 走了缓存~~~,NewsApi,newsList

```

2. 管理台

elementui讲解: <https://element.eleme.cn/>

2.1 需求

新闻管理, 需要后台进行添加, 修改等操作, 故需要一个管理台模块, 让网站运营人员可以便捷的进行一些运营管理操作, 管理台不止可以进行新闻管理, 后续实现题目上传, 订单管理, 营销管理等等都需要在后台进行实现

2.2 管理台登录功能

2.2.1 数据库设计

保存		添加字段		插入字段		删除字段		主键		上移 下移	
字段		索引	外键	触发器	选项	注释	SQL 预览				
名		类型			长度	小数点	不是 n	虚拟	键	注释	
▶ id		int			0	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>		1	主键id
username		varchar			45	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>			用户名
password		varchar			255	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>			密码

2.2.2 步骤分析

1. 用户输入用户名, 密码进行登录
2. 使用spring security进行用户的校验, 同时实现用户的权限管理

2.2.3 spring security 介绍

Spring Security 是一个功能强大且高度可定制的身份验证和访问控制框架。它是用于保护基于Spring的应用程序的实用标准。

Spring Security 致力于为Java应用程序提供身份验证和授权。与所有Spring项目一样, Spring Security的真正强大之处在于可以轻松扩展以满足自定义要求

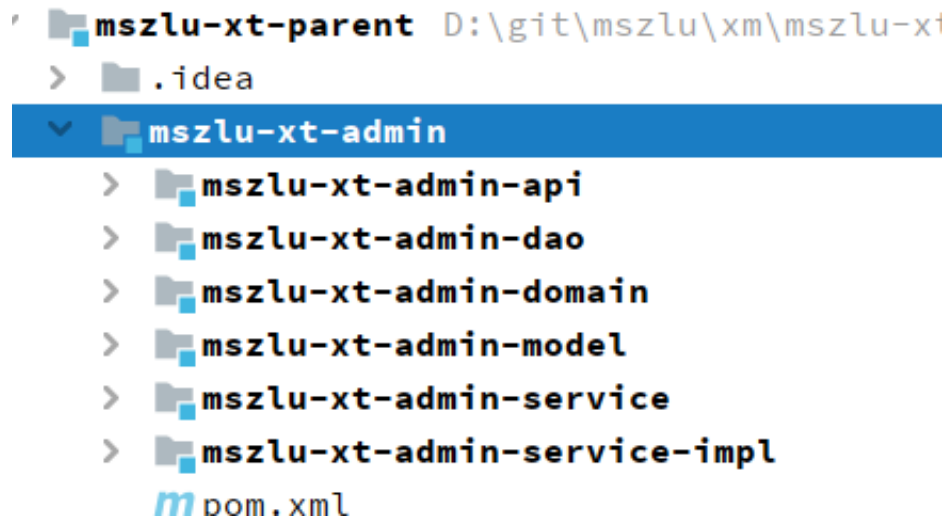
特性:

- 全面和可扩展的身份验证和授权支持
- 防止会话固定、点击劫持、跨网站请求伪造等攻击
- Servlet API集成

- 与Spring Web MVC的可选集成
- ...

2.2.4 编码

1. 新建admin模块，按之前的格式创建api，service，domain，dao，model等子模块



2. 导入依赖和web工程的依赖一致，多添加spring-security的支持

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-security</artifactId>
</dependency>
```

```
#application.properties
server.port=8228
server.servlet.session.timeout=604800s
spring.application.admin.enabled=false
spring.jmx.enabled=false
logging.level.com.mszlu.xt.admin.dao=debug
server.servlet.context-path=/lzadmin

##mybatisplus
mybatis-plus.global-config.db-config.table-prefix=t_
mybatis-plus.mapper-locations=classpath:mybatis/mapper/*.xml
mybatis-plus.configuration.log-impl=org.apache.ibatis.logging.stdout.StdOutImpl

#freemarker
spring.freemarker.check-template-location=true
spring.freemarker.charset=UTF-8
spring.freemarker.content-type=text/html; charset=utf-8
spring.freemarker.expose-request-attributes=true
```

```
spring.freemarker.expose-session-attributes=true
spring.freemarker.request-context-attribute=request

spring.profiles.active=local
#spring jackson
spring.jackson.time-zone=GMT+8
spring.servlet.multipart.max-file-size=30MB
spring.servlet.multipart.max-request-size=30MB
```

```
#application-local.properties
server.port=8228
#server.use-forward-headers=true
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/smart_pig?
useUnicode=true&characterEncoding=utf-8&serverTimezone=UTC
spring.datasource.username=root
spring.datasource.password=root

#redis 配置
spring.redis.host=localhost
spring.redis.port=6379
```

3. 在api模块的resources下新建static文件夹，将前端页面文件放入其中, 将web工程的mybatis等配置文件copy过来
4. 在api中创建security的配置文件

```
package com.msclu.xt.admin.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import
org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;
import
org.springframework.security.config.annotation.web.builders.HttpSecurity;
import
org.springframework.security.config.annotation.web.builders.WebSecurity;
import
org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;
```

```

import
org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import
org.springframework.security.crypto.password.PasswordEncoder;

@Configuration
public class SpringSecurityConfig extends
WebSecurityConfigurerAdapter {

    @Bean
    public PasswordEncoder getPasswordEncoder(){
        return new BCryptPasswordEncoder();
    }
    @Override
    protected void configure(HttpSecurity http) throws Exception {

        http.authorizeRequests()
            .antMatchers("/login.html").permitAll()
            .antMatchers("/css/**").permitAll()
            .antMatchers("/js/**").permitAll()
            .antMatchers("/plugins/**").permitAll()
            .anyRequest().authenticated()
            .and().headers().frameOptions().disable()
            .and()
            .formLogin()
            .defaultSuccessUrl("/pages/main.html")
            .permitAll()
            .and().logout()
            .and().csrf().disable()
            ;
//        super.configure(http);
    }

    @Override
    protected void configure(AuthenticationManagerBuilder auth)
throws Exception {
        super.configure(auth);
    }

    @Override
    public void configure(WebSecurity web) throws Exception {
        super.configure(web);
    }
}

```

5. 创建AdminUserService，根据用户名查询对应的用户信息

```
package com.mszlu.xt.admin.dao.data;

import lombok.Data;

@Data
public class AdminUser {

    private Long id;
    private String username;
    private String password;
}
```

```
package com.mszlu.xt.admin.service;

import com.mszlu.common.model.CallResult;

public interface AdminUserService {

    CallResult findUser(String username);
}
```

```
package com.mszlu.xt.admin.service.impl;

import com.mszlu.common.model.CallResult;
import com.mszlu.common.service.AbstractTemplateAction;
import com.mszlu.xt.admin.domain.AdminUserDomain;
import com.mszlu.xt.admin.domain.repository.AdminUserDomainRepository;
import com.mszlu.xt.admin.model.param.AdminUserParam;
import com.mszlu.xt.admin.service.AdminUserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
```

```

public class AdminUserServiceImpl extends AbstractService
implements AdminUserService {

    @Autowired
    private AdminUserDomainRepository adminUserDomainRepository;

    @Override
    public CallResult findUser(String username) {
        AdminUserParam adminUserParam = new AdminUserParam();
        adminUserParam.setUsername(username);
        AdminUserDomain adminUserDomain =
this.adminUserDomainRepository.createDomain(adminUserParam);
        return this.serviceTemplate.executeQuery(new
AbstractTemplateAction<Object>() {
            @Override
            public CallResult<Object> doAction() {
                return adminUserDomain.findUserByUsername();
            }
        });
    }
}

```

```

package com.mszlu.xt.admin.dao;

import com.baomidou.mybatisplus.core.mapper.BaseMapper;
import com.mszlu.xt.admin.dao.data.AdminUser;

public interface AdminUserMapper extends BaseMapper<AdminUser> {
}

```

```

package com.mszlu.xt.admin.domain.repository;

import
com.baomidou.mybatisplus.core.conditions.query.LambdaQueryWrapper;
import com.mszlu.xt.admin.dao.AdminUserMapper;
import com.mszlu.xt.admin.dao.data.AdminUser;
import com.mszlu.xt.admin.domain.AdminUserDomain;
import com.mszlu.xt.admin.model.param.AdminUserParam;
import org.springframework.beans.factory.annotation.Autowired;

```



```

import org.springframework.stereotype.Component;

@Component
public class AdminUserDomainRepository {

    @Autowired
    private AdminUserMapper adminUserMapper;

    public AdminUserDomain createDomain(AdminUserParam
adminUserParam) {
        return new AdminUserDomain(this,adminUserParam);
    }

    public AdminUser findUserByUsername(String username) {
        LambdaQueryWrapper<AdminUser> queryWrapper = new
LambdaQueryWrapper<>();

        queryWrapper.eq(AdminUser::getUsername,username).last("limit 1");
        return adminUserMapper.selectOne(queryWrapper);
    }
}

```

```

package com.mszlu.xt.admin.domain;

import com.mszlu.common.model.CallResult;
import com.mszlu.xt.admin.dao.data.AdminUser;
import
com.mszlu.xt.admin.domain.repository.AdminUserDomainRepository;
import com.mszlu.xt.admin.model.param.AdminUserParam;

public class AdminUserDomain {

    private AdminUserDomainRepository adminUserDomainRepository;
    private AdminUserParam adminUserParam;

    public AdminUserDomain(AdminUserDomainRepository
adminUserDomainRepository, AdminUserParam adminUserParam) {
        this.adminUserDomainRepository = adminUserDomainRepository;
        this.adminUserParam = adminUserParam;
    }

    public CallResult<Object> findUserByUsername() {

```

```

        AdminUser adminUser =
adminUserDomainRepository.findUserByUsername(this.adminUserParam.ge
tUsername());
        return CallResult.success(adminUser);
    }
}

```

6. 创建SpringSecurityService 实现SpringSecurity提供的UserDetailsService接口，实现用户认证

```

package com.mszlu.xt.admin.security;

import com.mszlu.common.model.CallResult;
import com.mszlu.xt.admin.dao.data.AdminUser;
import com.mszlu.xt.admin.service.AdminUserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.userdetails.User;
import org.springframework.security.core.userdetails.UserDetails;
import
org.springframework.security.core.userdetails.UserDetailsService;
import
org.springframework.security.core.userdetails.UsernameNotFoundException;
import
org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.stereotype.Component;

import java.util.ArrayList;
import java.util.List;

@Component
public class SpringSecurityService implements UserDetailsService {
    @Autowired
    private AdminUserService adminUserService;
    @Override
    public UserDetails loadUserByUsername(String username) throws
UsernameNotFoundException {
        List<GrantedAuthority> list = new ArrayList<>();
        CallResult callResult =
adminUserService.findUser(username);
        if (!callResult.isSuccess()){

```

```

        throw new UsernameNotFoundException("用户名不存在");
    }
    AdminUser adminUser = (AdminUser) callResult.getResult();
    return new User(username, adminUser.getPassword(), list);
}

public static void main(String[] args) {
    System.out.println(new
BCryptPasswordEncoder().encode("admin"));
}
}

```


7. 测试

2.3 管理平台新闻管理

2.3.1 需求

web端向用户展示了新闻列表, 自然需要运营人员在管理平台进行新闻的管理。

添加, 删除, 修改等。

字段	索引	外键	触发器	选项	注释	SQL 预览					
名						长度	小数点	不是 n	虚拟	键	注释
id						0	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>		id
title						255	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>		新闻题目
summary						255	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>		新闻概述
image_url						255	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>		封面图片地址
content						0	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>		内容
tab						0	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>		标签
create_time						0	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>		创建时间
author						255	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>		作者
n_status						0	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>		状态 0 有效 1 无效

```

package com.mszlu.xt.admin.dao.data;

import com.baomidou.mybatisplus.annotation.TableField;
import lombok.Data;

/**

```

```

* 新闻
*/
@Data
public class News {
    private Long id;
    private String title;
    private String summary;
    private String imageUrl;
    private String content;
    /**
     * 1 题库 2 升学 3 其他
     */
    private Integer tab;
    private Long createTime;
    private String author;
    /**
     * 0 正常 1 删除
     */
    @TableField("n_status")
    private Integer status;
}

```

2.3.2 步骤分析

1. 新建页面，news.html放入pages目录
2. 写对应的admin接口，实现新闻的分页查询，添加，修改，删除功能。

2.3.3 编码

1. 将资料中的news.html拷贝到pages目录
2. 新建NewsController，按照页面的接口定义，实现对应的接口

```

package com.msztu.xt.admin.model.param;

import lombok.Data;

@Data
public class NewsParam {
    private int page = 1;
    private int pageSize = 20;
    private Long id;
    private Integer tab;
}

```

```

private String title;
private String summary;
private String imageUrl;
private String content;
private Long createTime;
private String author;
/**
 * 0 正常 1 删除
 */
private Integer status;
}

```

```

package com.mszlu.xt.admin.controller;

import com.mszlu.common.model.CallResult;
import com.mszlu.xt.admin.model.param.NewsParam;
import com.mszlu.xt.admin.service.NewsService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

/**
 * @author Jarno
 */
@RestController
@RequestMapping("news")
public class AdminNewsController {

    @Autowired
    private NewsService newsService;

    @RequestMapping(value = "save")
    public CallResult save(@RequestBody NewsParam newsParam){
        return newsService.save(newsParam);
    }

    @RequestMapping(value = "update")
    public CallResult update(@RequestBody NewsParam newsParam){

```

```

        return newsService.update(newsParam);
    }

    @RequestMapping(value = "findNewsById")
    public CallResult findNewsById(@RequestBody NewsParam
newsParam){
        return newsService.findNewsById(newsParam);
    }

    @RequestMapping(value = "findPage")
    public CallResult findPage(@RequestBody NewsParam newsParam){
        return newsService.findPage(newsParam);
    }
}

```

3. 实现NewsService

```

package com.mszlu.xt.admin.service;

import com.mszlu.common.model.CallResult;
import com.mszlu.xt.admin.model.param.NewsParam;

public interface NewsService {

    CallResult save(NewsParam newsParam);

    CallResult update(NewsParam newsParam);

    CallResult findNewsById(NewsParam newsParam);

    CallResult findPage(NewsParam newsParam);
}

```

```

package com.mszlu.xt.admin.service.impl;

import com.mszlu.common.model.CallResult;
import com.mszlu.common.service.AbstractTemplateAction;
import com.mszlu.xt.admin.domain.NewsDomain;
import com.mszlu.xt.admin.domain.repository.NewsDomainRepository;

```

```

import com.mszlu.xt.admin.model.param.NewsParam;
import com.mszlu.xt.admin.service.NewsService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class NewsServiceImpl extends AbstractService implements
NewsService {

    @Autowired
    private NewsDomainRepository newsDomainRepository;

    @Override
    public CallResult save(NewsParam newsParam) {
        NewsDomain newsDomain =
this.newsDomainRepository.createDomain(newsParam);
        return this.serviceTemplate.execute(new
AbstractTemplateAction<Object>() {
            @Override
            public CallResult<Object> doAction() {
                return newsDomain.save();
            }
        });
    }

    @Override
    public CallResult update(NewsParam newsParam) {
        NewsDomain newsDomain =
this.newsDomainRepository.createDomain(newsParam);
        return this.serviceTemplate.execute(new
AbstractTemplateAction<Object>() {
            @Override
            public CallResult<Object> doAction() {
                return newsDomain.update();
            }
        });
    }

    @Override
    public CallResult findNewsById(NewsParam newsParam) {
        NewsDomain newsDomain =
this.newsDomainRepository.createDomain(newsParam);
        return this.serviceTemplate.execute(new
AbstractTemplateAction<Object>() {

```

```

        @Override
        public CallResult<Object> doAction() {
            return newsDomain.findNewsById();
        }
    });
}

@Override
public CallResult findPage(NewsParam newsParam) {
    NewsDomain newsDomain =
this.newsDomainRepository.createDomain(newsParam);
    return this.serviceTemplate.execute(new
AbstractTemplateAction<Object>() {
        @Override
        public CallResult<Object> doAction() {
            return newsDomain.findPage();
        }
    });
}
}
}

```

```

package com.mszlu.xt.admin.domain;

import com.baomidou.mybatisplus.extension.plugins.pagination.Page;
import com.mszlu.common.model.CallResult;
import com.mszlu.common.model.ListModel;
import com.mszlu.xt.admin.dao.data.News;
import com.mszlu.xt.admin.domain.repository.NewsDomainRepository;
import com.mszlu.xt.admin.model.NewsModel;
import com.mszlu.xt.admin.model.enums.NewsTab;
import com.mszlu.xt.admin.model.param.NewsParam;
import org.apache.commons.lang3.StringUtils;
import org.joda.time.DateTime;
import org.springframework.beans.BeanUtils;

import java.util.ArrayList;
import java.util.List;

public class NewsDomain {
    private NewsDomainRepository newsDomainRepository;
    private NewsParam newsParam;
}

```



```

    public NewsDomain(NewsDomainRepository newsDomainRepository,
NewsParam newsParam) {
        this.newsDomainRepository = newsDomainRepository;
        this.newsParam = newsParam;
    }

    public CallResult<Object> update() {
        Long id = this.newsParam.getId();
        News news = this.newsDomainRepository.findNews(id);
        if (news != null){
            String author = this.newsParam.getAuthor();
            if (!StringUtils.isEmpty(author)){
                news.setAuthor(author);
            }
            String title = this.newsParam.getTitle();
            if (!StringUtils.isEmpty(title)){
                news.setTitle(title);
            }
            String content = this.newsParam.getContent();
            if (!StringUtils.isEmpty(content)){
                news.setContent(content);
            }
            String summary = this.newsParam.getSummary();
            if (!StringUtils.isEmpty(summary)){
                news.setSummary(summary);
            }
            String imageUrl = this.newsParam.getImageUrl();
            if (!StringUtils.isEmpty(imageUrl)){
                news.setImageUrl(imageUrl);
            }
            Integer tab = this.newsParam.getTab();
            if (tab != null && NewsTab.valueOfCode(tab) != null){
                news.setTab(tab);
            }
            Integer status = this.newsParam.getStatus();
            if (status != null){
                news.setStatus(status);
            }
        }
        this.newsDomainRepository.updateNews(news);
        return CallResult.success();
    }

    public CallResult<Object> save() {

```

```

News news = new News();
news.setAuthor(this.newsParam.getAuthor());
news.setContent(this.newsParam.getContent());
news.setCreateTime(System.currentTimeMillis());
news.setSummary(this.newsParam.getSummary());
news.setImageUrl(this.newsParam.getImageUrl());
news.setTitle(this.newsParam.getTitle());
Integer status = this.newsParam.getStatus();
if (status == null){
    status = 0;
}
news.setStatus(status);
news.setTab(this.newsParam.getTab());
this.newsDomainRepository.save(news);
return CallResult.success();
}

public CallResult<Object> findNewsById() {
    News news =
this.newsDomainRepository.findNews(this.newsParam.getId());
    return CallResult.success(news);
}

public CallResult<Object> findPage() {
    int page = this.newsParam.getPage();
    int pageSize = this.newsParam.getPageSize();
    Page<News> newsPage =
this.newsDomainRepository.findAll(page, pageSize);
    ListModel<NewsModel> listModel = new ListModel<>();
    int total = (int) newsPage.getTotal();
    listModel.setTotal(total);
    List<News> result = newsPage.getRecords();
    List<NewsModel> newsModelList = new ArrayList<>();
    for (News news : result) {
        NewsModel newsModel = new NewsModel();
        BeanUtils.copyProperties(news, newsModel);
        newsModel.setCreateTimeStr(new
DateTime(news.getCreateTime()).toString("yyyy-MM-dd HH:mm:ss"));
        newsModelList.add(newsModel);
    }
    listModel.setList(newsModelList);
    listModel.setPage(page);
    listModel.setPageCount((int) newsPage.getPages());
    return CallResult.success(listModel);
}

```

```
}  
}
```

```
package com.mszlu.xt.admin.domain.repository;  
  
import  
com.baomidou.mybatisplus.core.conditions.query.LambdaQueryWrapper;  
import com.baomidou.mybatisplus.extension.plugins.pagination.Page;  
import com.mszlu.xt.admin.dao.NewsMapper;  
import com.mszlu.xt.admin.dao.data.News;  
import com.mszlu.xt.admin.domain.NewsDomain;  
import com.mszlu.xt.admin.model.param.NewsParam;  
import org.springframework.stereotype.Component;  
  
import javax.annotation.Resource;  
  
@Component  
public class NewsDomainRepository {  
    @Resource  
    private NewsMapper newsMapper;  
  
    public NewsDomain createDomain(NewsParam newsParam) {  
        return new NewsDomain(this, newsParam);  
    }  
  
    public News findNews(Long id) {  
        return newsMapper.selectById(id);  
    }  
  
    public void save(News news) {  
        newsMapper.insert(news);  
    }  
  
    public void updateNews(News news) {  
        newsMapper.updateById(news);  
    }  
  
    public Page<News> findAll(int page, int pageSize) {  
        LambdaQueryWrapper<News> queryWrapper = new  
LambdaQueryWrapper<>();  
        Page<News> page1 = new Page<>(page, pageSize);  
        queryWrapper.orderByDesc(News::getCreateTime);  
        return newsMapper.selectPage(page1, queryWrapper);  
    }  
}
```

```
}  
}
```

```
package com.mszlu.common.model;  
  
import java.io.Serializable;  
import java.util.ArrayList;  
import java.util.List;  
  
/**  
 * @author Jarno  
 */  
public class ListModel<T> implements Serializable {  
  
    private int pageCount;  
  
    private int page;  
  
    private int total;  
  
    private List<T> list;  
  
    public int getPageCount() {  
        return pageCount;  
    }  
  
    public void setPageCount(int pageCount) {  
        this.pageCount = pageCount;  
    }  
  
    public List<T> getList() {  
        return list;  
    }  
  
    public void setList(List<T> list) {  
        this.list = list;  
    }  
  
    public int getPage() {  
        return page;  
    }  
  
    public void setPage(int page) {
```

```

        this.page = page;
    }

    public ListModel<T> initNull() {
        ListModel<T> listModel = new ListModel<T>();
        listModel.setList(new ArrayList<T>());
        listModel.setPage(1);
        listModel.setPageCount(1);
        return listModel;
    }

    public int getTotal() {
        return total;
    }

    public void setTotal(int total) {
        this.total = total;
    }
}

```

```

package com.mszlu.xt.admin.model;

import lombok.Data;

/**
 * 新闻
 */
@Data
public class NewsModel {
    private Long id;
    private String title;
    private String summary;
    private String imageUrl;
    private String content;
    /**
     * 1 题库 2 升学 3 其他
     */
    private Integer tab;
    private Long createTime;
    private String author;
    /**
     * 0 正常 1 删除
     */
}

```

```
private Integer status;

private String createTimeStr;

}
```

```
package com.mszlu.xt.admin.model.enums;

import java.util.HashMap;
import java.util.Map;

/**
 * @author Jarno
 */
public enum NewsTab {
    /**
     * look name
     */
    TK(1, "tk"),
    SX(2, "sx"),
    OTHER(3, "OTHER");

    private static final Map<Integer, NewsTab> CODE_MAP = new
    HashMap<>(3);

    static{
        for(NewsTab topicType: values()){
            CODE_MAP.put(topicType.getCode(), topicType);
        }
    }

    /**
     * 根据code获取枚举值
     * @param code
     * @return
     */
    public static NewsTab valueOfCode(int code){
        return CODE_MAP.get(code);
    }

    private int code;
    private String msg;

    NewsTab(int code, String msg) {
```

```

        this.code = code;
        this.msg = msg;
    }

    public int getCode() {
        return code;
    }

    public void setCode(int code) {
        this.code = code;
    }

    public String getMsg() {
        return msg;
    }

    public void setMsg(String msg) {
        this.msg = msg;
    }
}

```

4. 测试

2.4 添加图片上传功能

elementui地址: <https://element.eleme.cn/#/zh-CN/component/installation>

其中文件上传组件的代码:

```

<el-upload
  class="avatar-uploader"
  action="https://jsonplaceholder.typicode.com/posts/"
  :show-file-list="false"
  :on-success="handleAvatarSuccess"
  :before-upload="beforeAvatarUpload">
  
  <i v-else class="el-icon-plus avatar-uploader-icon"></i>
</el-upload>

<script>

```

```
export default {
  data() {
    return {
      imageUrl: ''
    };
  },
  methods: {
    handleAvatarSuccess(res, file) {
      this.imageUrl = URL.createObjectURL(file.raw);
    },
    beforeAvatarUpload(file) {
      const isJPG = file.type === 'image/jpeg';
      const isLt2M = file.size / 1024 / 1024 < 2;

      if (!isJPG) {
        this.$message.error('上传头像图片只能是 JPG 格式!');
      }
      if (!isLt2M) {
        this.$message.error('上传头像图片大小不能超过 2MB!');
      }
      return isJPG && isLt2M;
    }
  }
}
</script>
```

2.4.1 修改新闻管理

新增和编辑弹窗修改：


```

<el-form-item label="图片地址" prop="imageUrl">
    <el-upload
        class="avatar-
uploader"
        action="/lzadmin/news/upload"
        name="newsImage"
        :show-file-
list="false"
        :on-
success="handleAvatarSuccess"
        :before-
upload="beforeAvatarUpload">
        
        <i v-else class="el-
icon-plus avatar-uploader-icon"></i>
    </el-upload>
</el-form-item>

```

```

methods: {
    handleAvatarSuccess(res, file){
        this.formData.imageUrl = res.result;
    },
    beforeAvatarUpload(){
        const isLt2M = file.size / 1024 / 1024 < 2;
        if (!isLt2M) {
            this.$message.error('上传图片大小不能超过 2MB!');
        }
        return isLt2M;
    }
}

```

2.4.2 写上传图片接口

使用七牛云上传

文档地址: <https://developer.qiniu.com/kodo/1239/java#upload-file>

依赖:

```
<dependency>
  <groupId>com.qiniu</groupId>
  <artifactId>qiniu-java-sdk</artifactId>
  <version>[7.7.0, 7.7.99]</version>
</dependency>
```

工具类:

```
package com.mszlu.common.utils;

import com.alibaba.fastjson.JSON;
import com.qiniu.common.QiniuException;
import com.qiniu.http.Response;
import com.qiniu.storage.Configuration;
import com.qiniu.storage.Region;
import com.qiniu.storage.UploadManager;
import com.qiniu.storage.model.DefaultPutRet;
import com.qiniu.util.Auth;

public class QiniuUtils {

    public static boolean upload(String accessKey,String
secretKey,String bucket,byte[] uploadBytes,String key){
        //构造一个带指定 Region 对象的配置类
        Configuration cfg = new Configuration(Region.huabei());
        //...其他参数参考类注释
        UploadManager uploadManager = new UploadManager(cfg);
        //...生成上传凭证, 然后准备上传
        //默认不指定key的情况下, 以文件内容的hash值作为文件名
        Auth auth = Auth.create(accessKey, secretKey);
        String upToken = auth.uploadToken(bucket);
        try {
            Response response = uploadManager.put(uploadBytes, key,
upToken);
            //解析上传成功的结果
            DefaultPutRet putRet =
JSON.parseObject(response.bodyString(), DefaultPutRet.class);
            return true;
        } catch (QiniuException ex) {
            Response r = ex.response;
            System.err.println(r.toString());
            try {
```

```

        System.err.println(r.bodyString());
    } catch (QiniuException ex2) {
        //ignore
    }
    return false;
}
}
}

```

配置类:

```

package com.mszlu.xt.admin.config;

import lombok.Data;
import
org.springframework.boot.context.properties.ConfigurationProperties;
import org.springframework.context.annotation.Configuration;

@Configuration
@ConfigurationProperties(prefix = "qiniu")
@Data
public class QiniuConfig {

    public String accessKey;

    public String accessSecret;

    public String bucket;
}

```

```

qiniu.accessKey=111
qiniu.accessSecret=222
qiniu.bucket=mszlu
qiniu.url=https://static.mszlu.com/

```

上传:

```

@Autowired
private QiniuConfig qiniuConfig;

```

```

    @PostMapping("upload")
    public CallResult upload(@RequestParam("newsImage")MultipartFile
file){
        String suffix =
StringUtils.substringAfterLast(file.getOriginalFilename(), ".");
        if (!"jpg,jpeg,png".contains(suffix)){
            return
CallResult.fail(BusinessCodeEnum.CHECK_PARAM_NO_RESULT.getCode(),"图片格
式错误");
        }
        DateTime dateTime = new DateTime();
        String fileName = "xt/"+ dateTime.toString("yyyy")+"/"+ +
dateTime.toString("mm")+"/"+UUID.randomUUID().toString()+". "+suffix;
        boolean upload = false;
        try {
            upload = QiniuUtils.upload(qiniuConfig.accessKey,
qiniuConfig.accessSecret, qiniuConfig.bucket, file.getBytes(),
fileName);
            if (upload){
                return CallResult.success(qiniuConfig.url+fileName);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
        return
CallResult.fail(BusinessCodeEnum.CHECK_PARAM_NO_RESULT.getCode(),"上传失
败");
    }

```

2.5 JRebel热部署

百度搜JRebel查询的资料: <https://blog.csdn.net/lianghecai52171314/article/details/105637251>