|  |
| --- |
| 上海子杰软件有限公司 |
| Java平台接口文档 |
| [第二次修订] |

|  |
| --- |
| 子杰软件  2017/8/7 |

目录

[1 登录 9](#_Toc493253504)

[1.1 登录 9](#_Toc493253505)

[1.1.1 后台登录接口 9](#_Toc493253506)

[1.1.2 前台登录接口 10](#_Toc493253507)

[1.1.3 产生验证码接口 11](#_Toc493253508)

[2 系统 12](#_Toc493253509)

[2.1 系统设置 12](#_Toc493253510)

[2.1.1 查找项目基本信息接口 12](#_Toc493253511)

[2.1.2 修改项目基本信息接口 12](#_Toc493253512)

[2.1.3 查找系统/扩展变量信息接口 12](#_Toc493253513)

[2.1.4 查找系统/扩展变量列表接口 13](#_Toc493253514)

[2.1.5 修改系统参数接口 14](#_Toc493253515)

[2.1.6 删除扩展变量接口 15](#_Toc493253516)

[2.1.7 增加扩展变量接口 16](#_Toc493253517)

[2.2 数据表与系统模型 17](#_Toc493253518)

[2.2.1 增加/修改数据表接口 17](#_Toc493253519)

[2.2.2 查找数据表信息接口 19](#_Toc493253520)

[2.2.3 查找数据表列表接口 20](#_Toc493253521)

[2.2.4 修改数据表顺序接口 22](#_Toc493253522)

[2.2.5 批量删除数据表接口 23](#_Toc493253523)

[2.2.6 查找字段信息接口 24](#_Toc493253524)

[2.2.7 字段查重接口 26](#_Toc493253525)

[2.2.8 查找字段列表接口 27](#_Toc493253526)

[2.2.9 删除字段接口 29](#_Toc493253527)

[2.2.10 增加/修改字段接口 30](#_Toc493253528)

[2.2.11 查找父表的ID和表名接口 34](#_Toc493253529)

[2.2.12 修改字段顺序接口 34](#_Toc493253530)

[2.2.13 显示html代码接口 35](#_Toc493253531)

[2.2.14 查找系统模型信息接口 37](#_Toc493253532)

[2.2.15 根据classid查系统模型接口 39](#_Toc493253533)

[2.2.16 查找系统模型列表接口 40](#_Toc493253534)

[2.2.17 删除系统模型接口 42](#_Toc493253535)

[2.2.18 增加/修改系统模型接口 43](#_Toc493253536)

[2.2.19 查询分表列表接口 44](#_Toc493253537)

[2.2.20 增加分表接口 45](#_Toc493253538)

[2.2.21 删除分表接口 46](#_Toc493253539)

[2.2.22 设置默认系统模型接口 47](#_Toc493253540)

[2.2.23 查询修改信息数据接口 48](#_Toc493253541)

[2.2.24 查询信息数据接口 49](#_Toc493253542)

[2.2.25 查询分表记录数接口 50](#_Toc493253543)

[2.2.26 查询信息列表接口 51](#_Toc493253544)

[2.2.27 删除信息接口 53](#_Toc493253545)

[2.2.28 操作信息接口 54](#_Toc493253546)

[2.2.29 获取不同基础表字段接口 55](#_Toc493253547)

[2.2.30 字段数据转移接口 56](#_Toc493253548)

[2.2.31 栏目（其他）转换信息接口 57](#_Toc493253549)

[2.2.32 信息转栏目(其他)接口 58](#_Toc493253550)

[2.2.33 批量信息归档接口 59](#_Toc493253551)

[2.2.34 批量信息修改接口 60](#_Toc493253552)

[2.2.35 信息转移/复制接口 62](#_Toc493253553)

[2.2.36 内容页html生成接口 63](#_Toc493253554)

[2.2.37 列表页html生成接口 64](#_Toc493253555)

[2.2.38 条件列表页html生成接口 65](#_Toc493253556)

[2.2.39 威客列表查询接口 66](#_Toc493253557)

[2.2.40 威客信息查询接口 67](#_Toc493253558)

[2.2.41 信息转换栏目接口 68](#_Toc493253559)

[2.2.42 信息转换其他接口 69](#_Toc493253560)

[2.2.43 信息转换人员接口 70](#_Toc493253561)

[2.2.44 增加阅读记录接口 72](#_Toc493253562)

[2.2.45 获取未读条数接口 73](#_Toc493253563)

[2.2.46 查询阅读记录接口 74](#_Toc493253564)

[2.2.47 查询阅读记录接口 75](#_Toc493253565)

[2.2.48 获取未读条数接口 77](#_Toc493253566)

[2.2.49 增加点击量接口 78](#_Toc493253567)

[2.2.50 获取信息点击量接口 79](#_Toc493253568)

[2.3 工作流模块 80](#_Toc493253569)

[2.3.1 增加工作流模型接口 80](#_Toc493253570)

[2.3.2 修改工作流模型 81](#_Toc493253571)

[2.3.3 修改工作流模型查找信息接口 82](#_Toc493253572)

[2.3.4 查找工作流模型列表接口 83](#_Toc493253573)

[2.3.5 删除工作流模型 85](#_Toc493253574)

[2.3.6 部署工作流模型接口 86](#_Toc493253575)

[2.3.7 执行（完成）一个流程任务接口 87](#_Toc493253576)

[2.3.8 启动（也可以叫申请或产生）一个新的流程接口 88](#_Toc493253577)

[2.3.9 查询某人已完成或未完成的任务 89](#_Toc493253578)

[2.3.10 查询流程实例信息 92](#_Toc493253579)

[2.3.11 查询流程实例的过程（任务列表） 93](#_Toc493253580)

[2.3.12 查询流程定义列表 95](#_Toc493253581)

[2.3.13 查询流程实例列表 96](#_Toc493253582)

[2.3.14 查找节点候选人名单接口 98](#_Toc493253583)

[2.3.15 拾取任务接口 99](#_Toc493253584)

[2.4 附件管理 100](#_Toc493253585)

[2.4.1 上传单个文件 100](#_Toc493253586)

[2.4.2 传附件和属性接口 101](#_Toc493253587)

[2.4.3 查找附件列表接口 103](#_Toc493253588)

[2.4.4 附件下载 105](#_Toc493253589)

[2.4.5 附件查看 106](#_Toc493253590)

[2.4.6 查询根级目录接口 107](#_Toc493253591)

[2.4.7 裁剪图片接口 108](#_Toc493253592)

[3 栏目 110](#_Toc493253593)

[3.1 栏目管理 110](#_Toc493253594)

[3.1.1 查看栏目列表接口： 110](#_Toc493253595)

[3.1.2 删除栏目接口 113](#_Toc493253596)

[3.1.3 增加/修改栏目接口 114](#_Toc493253597)

[3.1.4 查找关联字段 115](#_Toc493253598)

[3.1.5 修改栏目顺序接口 116](#_Toc493253599)

[3.1.6 查询绑定模板的栏目 117](#_Toc493253600)

[3.1.7 栏目转换信息接口 120](#_Toc493253601)

[4 模版 121](#_Toc493253602)

[4.1 列表模版 121](#_Toc493253603)

[4.1.1 查找列表模版列表接口 121](#_Toc493253604)

[4.1.2 查找列表模版信息接口 123](#_Toc493253605)

[4.1.3 删除列表模版接口 124](#_Toc493253606)

[4.1.4 增加/修改列表模版接口 125](#_Toc493253607)

[4.2 内容模版 126](#_Toc493253608)

[4.2.1 分查找内容模版列表接口 126](#_Toc493253609)

[4.2.2 查找内容模版信息接口 128](#_Toc493253610)

[4.2.3 删除内容模版接口 129](#_Toc493253611)

[4.2.4 增加/修改内容模版接口 130](#_Toc493253612)

[4.3 封面模版 131](#_Toc493253613)

[4.3.1 查找封面模版列表接口 131](#_Toc493253614)

[4.3.2 查找封面模版信息接口 133](#_Toc493253615)

[4.3.3 删除封面模版接口 134](#_Toc493253616)

[4.3.4 增加/修改封面模版接口 135](#_Toc493253617)

[5 用户 136](#_Toc493253618)

[5.1 机构管理 136](#_Toc493253619)

[5.1.1 查找机构/部门列表接口 136](#_Toc493253620)

[5.1.2 查找机构/部门及其所有子孙部门列表信息接口 138](#_Toc493253621)

[5.1.3 查找子机构/部门列表接口 （与4.1.1功能重复） 139](#_Toc493253622)

[5.1.4 查找机构/部门信息接口 + 141](#_Toc493253623)

[5.1.5 修改机构/部门顺序接口 142](#_Toc493253624)

[5.1.6 删除机构/部门接口 143](#_Toc493253625)

[5.1.7 配置机构角色接口（机构前端、后台角色不同表，做成同一个接口，传一个判断参数） 144](#_Toc493253626)

[5.1.8 删除机构角色接口（机构前端、后台角色不同表，做成同一个接口，传一个判断参数） 145](#_Toc493253627)

[5.1.9 查找已配置的机构或部门角色列表 (前后台共用) （机构前端、后台角色不同表，做成同一个接口，传一个判断参数） 146](#_Toc493253628)

[5.1.10 查找未配置的机构或部门角色列表 (前后台共用) （感觉没用） 147](#_Toc493253629)

[5.1.11 查找机构/部门下员工列表接口（前端人员） 148](#_Toc493253630)

[5.1.12 查找前端员工信息接口 + 149](#_Toc493253631)

[5.1.13 查找未配置的部门人员列表（感觉没用） 150](#_Toc493253632)

[5.1.14 删除员工接口（前端人员） 152](#_Toc493253633)

[5.1.15 删除员工角色接口（前端、后台共用） 153](#_Toc493253634)

[5.1.16 配置员工角色接口（前端、后台共用，传一个判断参数是否是管理员角色） 154](#_Toc493253635)

[5.1.17 查看人员机构/部门前端角色列表 155](#_Toc493253636)

[5.1.18 查看人员机构/部门后台角色列表 156](#_Toc493253637)

[5.1.19 删除员工标签接口 156](#_Toc493253638)

[5.1.20 配置员工标签接口 157](#_Toc493253639)

[5.1.21 删除机构数据词典 158](#_Toc493253640)

[5.1.22 配置机构数据词典 159](#_Toc493253641)

[5.1.23 导入机构数据词典 160](#_Toc493253642)

[5.1.24 导出机构数据词典 160](#_Toc493253643)

[5.1.25 更新机构数据词典 160](#_Toc493253644)

[5.1.26 查询机构未配置的数据词典列表（感觉没用） 160](#_Toc493253645)

[5.1.27 查询机构已配置的数据词典列表 161](#_Toc493253646)

[5.1.28 增加/修改机构接口 + 162](#_Toc493253647)

[5.1.29 增加/修改员工 + 163](#_Toc493253648)

[5.1.30 批量导入员工接口 165](#_Toc493253649)

[5.1.31 批量导出员工接口 166](#_Toc493253650)

[5.1.32 机构/部门信息转化接口 167](#_Toc493253651)

[5.1.33 查看员工前端/后台登录记录接口 167](#_Toc493253652)

[5.2 数据词典管理 167](#_Toc493253653)

[5.2.1 查找数据词典列表接口 167](#_Toc493253654)

[5.2.2 查找数据词典信息接口 169](#_Toc493253655)

[5.2.3 删除数据词典接口 170](#_Toc493253656)

[5.2.4 修改数据词典顺序接口 171](#_Toc493253657)

[5.2.5 导入数据词典到excel接口 172](#_Toc493253658)

[5.2.6 导出数据词典到excel接口 172](#_Toc493253659)

[5.2.7 生成Json接口 173](#_Toc493253660)

[5.2.8 增加/修改数据词典接口 174](#_Toc493253661)

[5.2.9 更新数据词典接口 176](#_Toc493253662)

[5.3 角色管理 176](#_Toc493253663)

[5.3.1 查找角色列表接口 176](#_Toc493253664)

[5.3.2 4.3.2 查找角色信息接口 178](#_Toc493253665)

[5.3.3 4.3.3 删除角色接口 179](#_Toc493253666)

[5.3.4 4.3.4 修改角色顺序接口 180](#_Toc493253667)

[5.3.5 4.3.5 增加/修改角色接口 181](#_Toc493253668)

[5.3.6 4.3.6 导入角色接口 183](#_Toc493253669)

[5.3.7 4.3.7 导出角色接口 183](#_Toc493253670)

[5.4 菜单管理 183](#_Toc493253671)

[4.4.12 查找菜单信息接口 183](#_Toc493253672)

[4.4.1 查找机构或部门前端/后台菜单列表 184](#_Toc493253673)

[4.4.2 新增/修改机构前端/后台菜单 186](#_Toc493253674)

[4.4.2 新增部门前端/后台菜单 186](#_Toc493253675)

[4.4.2 删除机构前端/后台菜单 186](#_Toc493253676)

[4.4.3 删除部门前端/后台菜单 186](#_Toc493253677)

[4.4.4 修改机构前端/后台菜单顺序 187](#_Toc493253678)

[4.4.5 查找员工前端菜单列表 187](#_Toc493253679)

[4.4.6 新增员工前端菜单列表 187](#_Toc493253680)

[4.4.7 删除员工前端菜单列表 187](#_Toc493253681)

[4.4.8 查找员工后台菜单列表 187](#_Toc493253682)

[4.4.9 新增员工后台菜单列表 187](#_Toc493253683)

[4.4.10 删除员工后台菜单列表 187](#_Toc493253684)

[4.4.13 查找机构或部门前端角色菜单列表 187](#_Toc493253685)

[4.4.14 新增机构或部门前端角色菜单列表 189](#_Toc493253686)

[4.4.15 删除机构或部门前端角色菜单列表 189](#_Toc493253687)

[4.4.16 查找机构或部门后台角色菜单列表 189](#_Toc493253688)

[4.4.15 删除机构或部门后台角色菜单列表 189](#_Toc493253689)

[4.4.14 新增机构或部门后台角色菜单列表 189](#_Toc493253690)

[4.4.3 查找人员前端菜单列表 189](#_Toc493253691)

[4.4.6 查询人员前端菜单及按钮接口 190](#_Toc493253692)

[4.4.7 查询人员前端私有菜单及按钮接口 190](#_Toc493253693)

[4.4.7 新增人员前端私有菜单及按钮接口 190](#_Toc493253694)

[4.4.7 删除人员前端私有菜单及按钮接口 190](#_Toc493253695)

[4.4.7 查询人员后台私有菜单及按钮接口 191](#_Toc493253696)

[4.4.7 新增人员后台私有菜单及按钮接口 191](#_Toc493253697)

[4.4.7 删除人员后台私有菜单及按钮接口 191](#_Toc493253698)

[4.4.7 查找机构后台按钮接口 191](#_Toc493253699)

[4.4.7 新增/修改机构后台按钮接口 191](#_Toc493253700)

[4.4.7 删除机构后台按钮接口 191](#_Toc493253701)

[4.4.7查找功能列表接口 191](#_Toc493253702)

[4.4.8 查找菜单功能信息接口 192](#_Toc493253703)

[4.4.9删除菜单接口 193](#_Toc493253704)

[5.5 标签管理 194](#_Toc493253705)

[4.5.1 查找标签列表接口 194](#_Toc493253706)

[4.5.2 查找标签信息接口 196](#_Toc493253707)

[4.5.3 删除标签接口 197](#_Toc493253708)

[4.5.4 导入标签 198](#_Toc493253709)

[4.5.5 导出标签 199](#_Toc493253710)

[4.5.6 增加/修改标签接口 199](#_Toc493253711)

[4.5.7 修改标签顺序接口 201](#_Toc493253712)

[5.6 按钮管理 202](#_Toc493253713)

[5.6.1 查询该机构角色的按钮配置的值 202](#_Toc493253714)

[5.6.2 更新机构角色的按钮配置的值 203](#_Toc493253715)

[5.6.3 获取按钮列表 203](#_Toc493253716)

[5.6.4 获取按钮详情 205](#_Toc493253717)

[5.6.5 修改和增加按钮 206](#_Toc493253718)

[5.6.6 删除按钮 206](#_Toc493253719)

[5.7 管理员管理 207](#_Toc493253720)

[5.7.1 增加管理员 207](#_Toc493253721)

[5.7.2 管理员匹配员工接口 208](#_Toc493253722)

[5.7.3 员工匹配管理员接口 208](#_Toc493253723)

[5.7.4 删除管理员接口 209](#_Toc493253724)

[5.7.5 修改管理员信息接口 210](#_Toc493253725)

[6 通用方法 210](#_Toc493253726)

[6.1 数据表与系统模型 210](#_Toc493253727)

[6.1.1 生成文件名方法 210](#_Toc493253728)

[6.1.2 生成页面信息方法 211](#_Toc493253729)

[6.1.3 生成列表信息方法 211](#_Toc493253730)

[6.1.4 刷新条件列表 212](#_Toc493253731)

[6.1.5 生成条件列表方法（单页） 212](#_Toc493253732)

[6.1.6 固有标签替换方法 213](#_Toc493253733)

[6.1.7 特殊标签处理方法 213](#_Toc493253734)

[6.1.8 归档方法 213](#_Toc493253735)

[6.1.9 信息操作方法 214](#_Toc493253736)

[6.1.10 其他转信息方法 215](#_Toc493253737)

[6.1.11 信息转其他方法 216](#_Toc493253738)

[6.1.12 查重接口 216](#_Toc493253739)

[6.1.13 查询列表接口（根据列表项动态） + 217](#_Toc493253740)

[6.1.14 查询信息数据接口（根据内容项动态） + 219](#_Toc493253741)

[6.2 系统基本设置 220](#_Toc493253742)

[6.2.1 获取系统参数方法 220](#_Toc493253743)

[6.2.2 修改系统参数方法 221](#_Toc493253744)

[7 对象 222](#_Toc493253745)

[7.1 系统用到的对象 222](#_Toc493253746)

[7.1.1 系统参数类对象 222](#_Toc493253747)

[7.1.2 模型数据表类对象 222](#_Toc493253748)

[7.1.3 字段类对象 223](#_Toc493253749)

[7.1.4 系统模型类对象 224](#_Toc493253750)

[7.1.5 基础字段类对象 225](#_Toc493253751)

[7.2 栏目用到的对象 226](#_Toc493253752)

[7.2.1 栏目主表对象 226](#_Toc493253753)

[7.2.2 栏目副表对象 227](#_Toc493253754)

[7.2.3 栏目副表 权限 228](#_Toc493253755)

[7.2.4 栏目副表 大字段 229](#_Toc493253756)

[7.3 模版类用到的对象 229](#_Toc493253757)

[7.3.1 列表模版类对象 229](#_Toc493253758)

[7.3.2 内容模版类对象 230](#_Toc493253759)

[7.3.3 封面模版类对象 230](#_Toc493253760)

[7.4 用户模块用到的对象 231](#_Toc493253761)

[7.4.1 机构类对象 231](#_Toc493253762)

[7.4.2 人员类对象 232](#_Toc493253763)

[7.4.3 角色类对象 232](#_Toc493253764)

[7.4.4 菜单类对象 233](#_Toc493253765)

[7.4.5 数据词典类对象 233](#_Toc493253766)

[7.4.6 标签类对象 234](#_Toc493253767)

[7.5 全局对象 234](#_Toc493253768)

[7.5.1 分页实体类对象 234](#_Toc493253769)

[7.5.2 返回信息类对象 235](#_Toc493253770)

[8 附录 235](#_Toc493253771)

[8.1 信息栏目转化字段对照表 235](#_Toc493253772)

1. 登录
   1. 登录
      1. 后台登录接口

**a.接口url:**

pub/admin/login.do

**b.接口说明:**

后台登陆检测的接口

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| login | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| usename | String | | 用户名 | | |
| password | String | | 用户密码 | | |
| ra | String | | 验证码（选填-视项目情况） | | |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "登陆成功", //信息

"flag": "1"， //标识

“token”:“12343487sjdfb”， //token值

"userId": "sdofhov" //用户id

}

}

**e.功能实现说明**

1. 从session中取出验证码，和上传验证码对比，不相等，则返回验证码不正确的信息。
2. 根据username查管理员数据库的数据，如果查不到，设置flag为0，msg为用户名不存在。获得系统参数看是一次加密，还是二次加密，对密码进行处理，在放到password
3. 如果用户名存在，把查到的数据和上传的password比较。
   1. 如果相等：
      1. 产生一个UUId,在对这个UUID字符化UUID1，更新到这个人员的token字段中。
      2. 设置flag为1，msg UUID1。
   2. 如果不相等，设置flag为0，msg为密码错误。
      1. 前台登录接口

**a.接口url:**

pub/admin/login1.do

**b.接口说明:**

前台登陆检测的接口

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| Login1 | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| username | String | | 用户名 | | |
| password | String | | 用户密码 | | |
| ra | String | | 验证码（选填-视项目情况） | | |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "登陆成功", //信息

"flag": "1"， //标识

“token”:“12343487sjdfb”， //token值

"userId": "sdofhov" //用户id

}

}

**e.功能实现说明**

1. 从session中取出验证码，和上传验证码对比，不相等，则返回验证码不正确的信息。
2. 根据username查人员表数据库的数据，如果查不到，设置flag为0，msg为用户名不存在。获得系统参数看是一次加密，还是二次加密，对密码进行处理，在放到password
3. 如果用户名存在，把查到的数据和上传的password比较。
   1. 如果相等：
      1. 产生一个UUId,在对这个UUID字符化UUID1，更新到这个人员的token字段中。
      2. 设置flag为1，msg UUID1。

如果不相等，设置flag为0，msg为密码错误。

* + 1. 产生验证码接口

**a.接口url:**

pub/admin/randomcode.do

**b.接口说明:**

向客服端输出验证码图片

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| randomcode | | Void | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
|  |  | |  | | |

**d.返回值示例与说明**

返回二进制文件

**e.功能实现说明**

1. 用java代码产生随机验证码图片发送到客服端
2. 系统
   1. 系统设置
      1. 查找项目基本信息接口
      2. 修改项目基本信息接口
      3. 查找系统/扩展变量信息接口

**a.接口url:**

system/setSystem/findSysparam.do

**b.接口说明:**

查找单个系统参数、扩展变量、安全参数、防火墙参数等。

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| findSysparam | | ZjxxSysparam  （[说明见6.1.1](#_6.1.1_系统参数类对象)） | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| title | String | | 参数英文名、扩展变量名 | | |

**d.返回值示例与说明**

{

"id": "1", //系统参数id

"title": "sitename", //参数英文名

"cname": "站点名称", //参数中文名

"content": "学习城测试", //内容

"type": "基本属性", //参数类型

"iscache": "1" //是否写入缓存

}

**e.功能实现说明**

调用util中获取系统参数方法

* + 1. 查找系统/扩展变量列表接口

**a.接口url:**

system/setSystem/findSysparamList.do

**b.接口说明:**

查找系统参数、扩展变量、安全参数、防火墙参数等的列表。

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| findSysparamList | | Info | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| type | String | | 用来标识参数类别，其值包括：  扩展变量、系统参数设置（基本设置、用户设置、文件设置、信息设置、消息推送、机构设置）、安全参数设置、网络防火墙设置。  以上均为固定值。 | | |
| page | String | | 当前页码 | | |
| num | String | | 每页显示多少条数据 | | |

**d.返回值示例与说明**{

"page": { //分页信息

"page": 2, //当前页

"pages": 10, //总页数

"lines": 100, //信息总条数

"pageSize": "10", //每页显示信息数

"pageStart": 1 //起始页

},

"total": 1, //当前页信息数

"infos": [ //信息

{

"id": "1", //系统参数id

"title": "sitename", //参数英文名

"cname": "站点名称", //参数中文名

"content": "学习城测试", //内容

"type": "基本属性", //参数类型

"iscache": "1" //是否写入缓存

}

]

}

**e.功能实现说明**

①检测参数typ当值不为空时加入查询条件。type使用精确查询（=）。

②检查page参数、为空则设置默认值1，检查num参数、为空则设置默认值30。

③根据page和num生成页码条件（limit）

④组合sql语句查询数据，根据返回值格式进行数据返回。

* + 1. 修改系统参数接口

**a.接口url:**

system/setSystem/updateSysparam.do

**b.接口说明:**

修改系统参数、扩展变量、安全参数、网站防火墙的数据。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| updateSysparam | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| zjxxSysparam | ZjxxSysparam  （[说明见6.1.1](#_6.1.1_系统参数类对象)） | 系统参数类对象 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "更新成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

调用util中修改系统参数方法

* + 1. 删除扩展变量接口

**a.接口url:**

system/setSystem/deleteSysparam.do

**b.接口说明:**

删除指定的扩展变量。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| deleteSysparam | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 扩展变量id |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "删除成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

①接收参数扩展变量id，在系统参数表中通过扩展变量id删除指定的扩展变量。

②根据缓存的key为id清除服务器上application的缓存。

③返回删除成功信息

* + 1. 增加扩展变量接口

**a.接口url:**

system/setSystem/addSysparam.do

**b.接口说明:**

增加扩展变量。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| addSysparam | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| zjxxSysparam | ZjxxSysparam  （[说明见6.1.1](#_6.1.1_系统参数类对象)） | 系统参数对象 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "增加成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

①传入的对象参数zjxxSysparam，方法中生成id并赋值给zjxxSysparam的id属性。

②执行insert方法将zjxxSysparam对象插入数据库系统参数表。

③返回更新成功信息

注：增加扩展变量时要查询变量名title有没有重复（在输入变量名到输入变量名失去焦点时开始判断），调用查重接口[5.1.12](#_5.1.12_查重接口)

* 1. 数据表与系统模型
     1. 增加/修改数据表接口

**a.接口url:**

system/tableModel/addOrUpdateTable.do

**b.接口说明:**

（1）如果传入zjxxEnewstable对象的tid为空，则为增加数据表接口。

（2）如果传入zjxxEnewstable对象的tid不为空，则为修改数据表接口。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| addOrUpdateTable | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| zjxxEnewstable | ZjxxEnewstable | 数据表对象 |
| originalTbname | String | 原始表名，修改表时用来区分修改前的表名和修改后的表名 |
| originalParenttb | String | 原始的父表，修改父表时用来区分修改前的父表和修改后的父表 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "增加成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

（1）如果zjxxEnewstable对象的id为空：

①传入参数zjxxEnewstable的对象、originalTbname、originalParenttb。生成UUID，设置zjxxEnewstable对象的id属性，将该对象设置为主表，设置默认顺序，设置主表表名和分表集合为空。

②获得zjxxEnewstable对象的父表parenttb，如果parenttb不为空，则:更新zjxx\_enewstable表中tbname为原parenttb的数据的子表集合字段。

③获得zjxxEnewstable对象的子表sontb，如果sontb不为空，则:更新zjxx\_enewstable表中tbname为sontb的数据的父表字段。

④根据zjxxEnewstable对象建立表a、归档表a\_gd、微客表a\_wk。

⑤为表a 、归档表a\_gd、微客表a\_wk添加系统默认字段，（设置默认字段的html）。

⑥a\_wk表额外增加weike\_id、modifyauthor、modifytime三个字段。将zjxxEnewstable对象数据插入zjxx\_enewstable表中，返回result。

（2）如果zjxxEnewstable对象的id不为空：

①接收参数zjxxEnewstable对象，则使用update()方法更新数据库中的数据。

如果originalTbname和zjxxEnewstable对象中的tbname不同，则执行②③④⑤：

②修改字段表中该表的字段信息，修改模型表中绑定该表的模型的tbname。

③修改数据库中对应的该表、该表的归档表、该表的微客表。

④找到该表的父表数据，修改其父表的子表集合。

⑤找到该表的子表集合，遍历子表集合，依次修改每一个子表的父表字段。

如果originalParenttb和zjxxEnewstable对象中的parenttb不同，则执行⑥⑦：

⑥通过originalParenttb找到原父表，修改原父表的子表集合sontb字段。

⑦通过parenttb找到更改后的父表，修改更改后的父表的子表集合sontb字段。

⑧返回result。

* + 1. 查找数据表信息接口

**a.接口url:**

system/tableModel/findTable.do

**b.接口说明:**

按条件查找数据表数据。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findTable | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| tid | String | 数据表id |
| tbname | String | 数据表表名 |

**d.返回值示例与说明**

{

"result": { //识别结果

"msg": "查询成功", //结果信息

"flag": "1" //1为成功，0为失败

}

"infos" :{

"datatbs": "zjxx\_t2\_1,", //分表集合

"ismaster": true, //是否是主表

"parenttb": "", //父表

"sontb": "zjxx\_t2\_son,", //子表集合

"tbname": "zjxx\_t2", //数据表名

"tbtype": 6, //表类型

"tid": "338055a8-15aa-4174-ac7b-0a65148125fb", //数据表ID

"tname": "t2", //数据表标识

"tsay": "t2", //备注

"tbs": "name,age,", //对应字段

"masterb": "", //主表名称

"torder": "1", //顺序

"yhid": "1", //优化方案ID

"orgid": "1" //机构

}

}

**e.功能实现说明**

1. 分别检测参数tbname、tid、若两个参数皆为空值、则返回“来源错误”信息。
2. 若tbname非空值、则拼接查询条件、使用精确查询（=）
3. 若tid非空值、则拼接查询条件、使用精确查询（=）
4. 根据生成的查询条件、查询对应数据库、找到数据后进行返回。
5. 若找不到数据则提示“无对应数据”信息
   * 1. 查找数据表列表接口

**a.接口url:**

system/tableModel/findTableList.do

**b.接口说明:**

查找数据表列表数据。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findTableList | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| method | String | 查找方式 1表示正选 2表示反选（选填） |
| tbname | String | 数据表表名（选填） |
| parenttb | String | 父表表名（选填） |
| page | String | 当前页码（选填） |
| num | String | 每页显示多少条数据（选填） |

**d.返回值示例与说明**

{

"page": { //分页信息

"page": 2, //当前页

"pages": 10, //总页数

"lines": 100, //信息总条数

"pageSize": "10", //每页显示信息数

"pageStart": 1 //起始页

},

"total": 1, //当前页信息数

"infos": [ //信息

{

"datatbs": "zjxx\_t2\_1,", //分表集合

"ismaster": true, //是否是主表

"parenttb": "", //父表

"sontb": "zjxx\_t2\_son,", //子表集合

"tbname": "zjxx\_t2", //数据表名

"tbtype": 6, //表类型

"tid": "338055a8-15aa-4174-ac7b-0a65148125fb", //数据表ID

"tname": "t2", //数据表标识

"tsay": "t2", //备注

"tbs": "name,age,", //对应字段

"masterb": "", //主表名称

"torder": "1", //顺序

"yhid": "1", //优化方案ID

"orgid": "1" //机构

}

]

}

**e.功能实现说明**

1. 检查page参数、为空则设置默认值1，检查num参数、为空则设置默认值30。
2. 检查method参数，为空则设置默认值1。
3. 检查tbname参数，若非空则根据method的值拼接查询条件:

method=1时，根据tbname进行模糊查询（like）、查找相似表名数据

method=2时，根据tbname进行反向查询（<>）、查找不为此名的表数据

1. 检查parenttb参数，为空则设置默认值“”
2. 根据parenttb拼接条件、精确查询（=）
3. 根据page和num生成页码条件（limit）
4. 根据生成的sql语句查询数据，返回数据。
   * 1. 修改数据表顺序接口

**a.接口url:**

system/tableModel/updateTableOrder.do

**b.接口说明:**

修改、批量修改数据表显示顺序。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| updateTableOrder | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| orders | String[] | 需要修改的顺序的数组 |
| tids | String[] | 数据表id的数组 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "修改成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

①接收参数tids和orders，遍历tids数组，每一条tids[i]对应一个orders[i],生成sql语句

“UPDATE zjxx\_enewstable SET torder = ‘orders[i]’ WHERE tid = ‘tids[i]’”

②执行sql语句更新数据库，返回result。

* + 1. 批量删除数据表接口

**a.接口url:**

system/tableModel/deleteTable.do

**b.接口说明:**

删除、批量删除数据表。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| deleteTable | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| tids | String[] | 数据表id数组 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "删除成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

①传入参数tids，遍历tids得到每个tid，使用finById(tid)找到该表数据，得到该表的父表数据，得到该表的表名tbname，修改其父表的子表集合。

②使用deleteByRecursion(tbname)方法递归删除该表下的系统模型、删除字段表中属于该表的字段、删除该表数据、删除数据库中的该表、删除该表的归档和微客表。

③得到该表的分表集合，遍历分表集合删除所有分表数据、删除分表字段、删除数据库中分表、删除分表的归档表和微客表。

④得到该表的子表集合，如果子表集合不为空，遍历子表集合，每次将一个子表的表名sontbname作为递归参数再次传入deleteByRecursion(sontbname)进行下一轮递归。

* + 1. 查找字段信息接口

**a.接口url:**

system/tableModel/findField.do

**b.接口说明:**

按条件查询字段数据。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findField | Infor  （[说明见6.1.3](#_6.1.3_字段类对象)） | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| tid | String | 数据表id |
| fid | String | 字段id |
| f | String | 字段名 |

**d.返回值示例与说明**

{infos:[ {

"f": "name", //字段名

"fform": "linkfield", //输入表单显示元素

"fhtml": "classid=<?=$classid?>", //输入表单替换html代码

"fid": "08a049db-2815-4004-91a3-c232c4bb0830", //字段ID

"flen": "255", //字段长度

"fname": "姓名", //字段标识

"ftype": "VARCHAR", //字段类型

"fvalue": "", //字段初始值

"fzs": "副标题", //注释

"isadd": false, //是否是系统字段

"iscj": false, //是否为采集项

"iskey": false, //是否为索引字段

"isonly": false, //字段值是否唯一

"ispage": false, //是否分页字段

"isshow": false, //是否显示字段

"issmalltext": false, //是否简介字段

"jsonstring": "{/"fformsize/":/"60/",/"fformwidth/":/"/",/"fformheight/":/"/",/"linkfieldtb/":/"/",/"linkfieldshow/":/"/",/"linkfieldval/":/"/",/"samedata/":/"1/",/"editorys/":/"0/"}", //

"qfhtml": "<input name=/"zjxxEnewsf.getF()/" type=/"text/" id=/"zjxxEnewsf.getF()/" value=/"<?=$ecmsfirstpost==1?/"/":ehtmlspecialchars(stripSlashes($r[zjxxEnewsf.getF()]))?>/" size=/"45/">/n <a onclick=/"window.open('ecmseditor/FileMain.php?type=2", //选项

"savetxt": false, //是否内容存文本

"tbdataf": 0, //是否为分表字段

"tbname": "zjxx\_t2", //数据表明

"tid": "338055a8-15aa-4174-ac7b-0a65148125fb" //数据表ID

}

]

}

**e.功能实现说明**

1. 检查参数、若fid为空且f和tid有一个及以上为空值时、返回“来源错误”信息。
2. 若fid非空值、则拼接查询条件、使用精确查询（=）
3. 若tid非空值、则拼接查询条件、使用精确查询（=）
4. 若f非空值、则拼接查询条件、使用精确查询（=）
5. 根据生成的查询条件、查询对应数据库、找到数据后进行返回。
6. 若找不到数据则提示“无对应数据”信息
7. 返回infor
   * 1. 字段查重接口

**a.接口url:**

system/tableModel/findSameField.do

**b.接口说明:**

添加和修改字段时，按条件查询字段数据判断是否重复。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findSameField | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| tbname | String | 数据表名 |
| fid | String | 字段id |
| f | String | 字段名 |

**d.返回值示例与说明**

{

"flag": "1", //标识：2重复，1成功 ，0出错；

" msg": "", //

}

**e.功能实现说明**

1. 检查参数、若f和tbname有一个及以上为空值时、返回“来源错误”信息。
2. 根据tbname找到其主表和所有分表并根据并f和tbnames拼接查询条件查找数据库。
3. 若fid为空值、判断查找结果是否为空，若存在则设置flag=2，不存在设置flag=1
4. 若fid非空值、判断查找结果是否为空，若存在则判断其fid是否等于fid，若相等，则设置flag=1，若不相等，则设置flag=2；若不存在设置flag=1；
5. 查询失败，设置flag=1，返回Result。
   * 1. 查找字段列表接口

**a.接口url:**

system/tableModel/findFieldList.do

**b.接口说明:**

查询字段数据列表。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findFieldList | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| tid | String | 数据表id |

**d.返回值示例与说明**

{

"infos": [ //信息

{

"f": "name", //字段名

"fform": "linkfield", //输入表单显示元素

"fhtml": "classid=<?=$classid?>", //输入表单替换html代码

"fid": "08a049db-2815-4004-91a3-c232c4bb0830", //字段ID

"flen": "255", //字段长度

"fname": "姓名", //字段标识

"ftype": "VARCHAR", //字段类型

"fvalue": "", //字段初始值

"fzs": "副标题", //注释

"isadd": false, //是否是系统字段

"iscj": false, //是否为采集项

"iskey": false, //是否为索引字段

"isonly": false, //字段值是否唯一

"ispage": false, //是否分页字段

"isshow": false, //是否显示字段

"issmalltext": false, //是否简介字段

"jsonstring": "{/"fformsize/":/"60/",/"fformwidth/":/"/",/"fformheight/":/"/",/"linkfieldtb/":/"/",/"linkfieldshow/":/"/",/"linkfieldval/":/"/",/"samedata/":/"1/",/"editorys/":/"0/"}", //

"qfhtml": "<input name=/"zjxxEnewsf.getF()/" type=/"text/" id=/"zjxxEnewsf.getF()/" value=/"<?=$ecmsfirstpost==1?/"/":ehtmlspecialchars(stripSlashes($r[zjxxEnewsf.getF()]))?>/" size=/"45/">/n <a onclick=/"window.open('ecmseditor/FileMain.php?type=2", //选项

"savetxt": false, //是否内容存文本

"tbdataf": 0, //是否为分表字段

"tbname": "zjxx\_t2", //数据表明

"tid": "338055a8-15aa-4174-ac7b-0a65148125fb" //数据表ID

}

],

[{…},{…}]

}

**e.功能实现说明**

①检查参数 tid ，若为空则提示返回“来源错误”

②根据tid查找对应数据表数据。将该表的表名、以及分表名存入一个数组tbnames[]

③遍历tbnames[]，依次根据各个表名再去获取对应字段的列表。存入返回值对象。

返回一个Info对象

* + 1. 删除字段接口

**a.接口url:**

system/tableModel/deleteField.do

**b.接口说明:**

删除字段。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| deleteField | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| fid | String | 字段id |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "删除成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

**e.功能实现说明**

① 通过fid找到对应的字段数据，得到字段名f、以及字段所属数据表tbname。

② 判断tbname是否为主表（根据字段数据的tid在模型数据表中查找该数据表记录， 查看记录的Masterb字段是否为空，若不为空，则则将Masterb赋给tbname）

③ 通过得到的tbname在系统模型表中查找所有tbname字段值为tbname的list对象集合，遍历该list集合。查找每一记录中的以上字段：mtemp、cj、enter、tempvar、searchvar、qenter、mustqenterf、listandf、listtempvar、canaddf、caneditf、orderf、uk、enterjson是否存在原字段信息，若存在则删除各字段中的原字段的信息（在删除时，注意连后面的“，”一并删除，可以使用replace替换为空），并更新系统模型表记录

④在数据库中删除tbname表中的字段f，删除对应归档表、微客表中的字段f。

⑤删除zjxx\_enewsf表中该符合该fid的字段数据。

* + 1. 增加/修改字段接口

**a.接口url:**

system/tableModel/addOrUpdateField.do

**b.接口说明:**

（1）如果传入参数zjxxEnewsf对象的fid为空，则为增加字段接口。

（2）如果zjxxEnewsf对象的fid不为空，则为修改字段、转移字段接口。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| addOrUpdateField | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| zjxxEnewsf | ZjxxEnewsf | 字段对象 |
| toTbname | String | 字段转移的目标表名 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "修改成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

（1）判断zjxxEnewsf对象的flen为空，则设置默认字段长度为255

（2）如果传入参数zjxxEnewsf对象的fid为空：为增加字段

①、生成uuid，并将其设置为字段的fid；

②、判断fthml是否为空或空字符串，若为空，则调用showHtml接口，设置其zjxxEnewsf的fhtml和qhtml的值。

③接收参数zjxxEnewsf对象，得到该字段的tbname。

④得到字段名、字段类型、字段长度、是否为空、是否为主键等信息。

⑤、判断zjxxEnewsf的ftype字段类型是否为TEXT，若是，则单独拼接sql语句（不考虑字段的值唯一和加索引属性）往数据库tbname表中添加该字段，往tbname的归档表和微客表中添加该字段。

⑥、判断zjxxEnewsf的ftype字段类型是否为Longtext、mediumtext，若是，则单独拼接对应的sql语句（不考虑字段的值唯一和加索引属性）往数据库tbname表中添加该字段，往tbname的归档表和微客表中添加该字段。

⑦、判断zjxxEnewsf的ftype字段类型是否有长度，分为有长度与无长度。

再判断字段的值isonly是否唯一,

根据不同的条件，拼接不同的sql语句。

分别给数据库tbname表中添加该字段，往tbname的归档表和微客表中添加该字段。

⑧、判断zjxxEnewsf的值不唯一 且 加索引 且 ftype的值不等于text、mediumtext、longtext时，给数据表对应的字段添加索引，生成sql语句。往数据库tbname表中添加该字段，往tbname的归档表和微客表中添加该字段。

~~③生成添加字段的sql语句，往数据库tbname表中添加该字段，往tbname的归档表和微客表中添加该字段。~~

⑨将zjxxEnewsf对象数据插入zjxx\_enewsf表中。

10.返回result。

（2）如果传入参数zjxxEnewsf对象的fid不为空：

①如果toTbname为空，则为修改字段

1. 通过zjxxEnewsf对象的fid在字段表中查找该字段的原数据oldzjxxEnewsf
2. 判断修改后的字段zjxxEnewsf是否有长度，拼接sql语句，修改数据表、归档表以及威客表中对应的字段
3. 若oldzjxxEnewsf的ftype不等于text、Longtext、mediumtext时，判断修改前字段oldzjxxEnewsf的属性是否存在值唯一或索引，若存在，则先拼接sql语句，去除字段的索引属性，修改数据表、归档表以及威客表中对应的字段
4. 若zjxxEnewsf的值唯一，且zjxxEnewsf的字段类型ftype不等于text、Longtext、mediumtext时，给字段添加唯一属性，修改数据表、归档表以及威客表中对应的字段
5. 若zjxxEnewsf的值不唯一且要加索引且zjxxEnewsf字段类型ftype不等于text、Longtext、mediumtext时，给字段添加索引属性，修改数据表、归档表以及威客表中对应的字段。

（以下6、7）为修改系统模型表操作

1. 再通过oldzjxxEnewsf的该字段的tbname，并判断其是否为主表（根据oldzjxxEnewsf的tid在模型数据表中查找该数据表记录，查看记录的Masterb字段是否为空，若不为空，则则将Masterb赋给tbname）
2. 通过得到的tbname在系统模型表中查找所有tbname字段值为tbname的list对象集合，遍历该list集合。查找每一记录中的以上字段：mtemp、cj、enter、tempvar、searchvar、qenter、mustqenterf、listandf、listtempvar、canaddf、caneditf、orderf、uk、enterjson是否存在原字段信息，若存在则根据zjxxEnewsf信息依次替换（可以使用replace替换），并更新系统模型表记录

②~~通过zjxxEnewsf得到tbname。~~

③更新zjxx\_enewsf表中的字段信息。

~~④修改数据库tbname表、tbname\_gd表和tbname\_wk表中对应字段的属性，返回result。~~

④、返回result；

⑤如果toTbname不为空，则：为转移字段

⑥通过zjxxEnewsf得到原数据表tbname。

⑦、设置zjxxEnewsf2为值为zjxxEnewsf，并修改其fid为空，再修改其tbname为toTbname，tid为相对与的值，调用添加字段接口，往数据库toTbname表中添加zjxxEnewsf字段，往数据库toTbname的归档表和微客表中添加zjxxEnewsf字段。

~~⑦往数据库toTbname表中添加zjxxEnewsf字段，往数据库toTbname的归档表和微客表中添加zjxxEnewsf字段。~~

（8、9）为转移字段时要修改的系统模型表信息

⑧ 通过zjxxEnewsf的该对象的tbname，并判断其是否为主表（根据zjxxEnewsf的tid在模型数据表中查找该数据表记录，查看记录的Masterb字段是否为空，若不为空，则则将Masterb赋给tbname）

⑨通过得到的tbname在系统模型表中查找所有tbname字段值为tbname的list对象集合，遍历该list集合。查找每一记录中的以上字段：mtemp、cj、enter、tempvar、searchvar、qenter、mustqenterf、listandf、listtempvar、canaddf、caneditf、orderf、uk、enterjson中是否存在zjxxEnewsf字段信息，若存在则根据totbname修改每个字段值中的tbname（可以使用replace替换），并更新系统模型表记录

⑨返回result。

\*字段转移逻辑中，前台人员还需调用信息转移接口和字段删除接口以完成全部逻辑。

* + 1. 查找父表的ID和表名接口

**a.接口url:**

system/tableModel/findparenttidtbname.do

**b.接口说明:**

（1）如果传入参数zjxxEnewsf对象的fid为空，则为增加字段接口。

（2）如果zjxxEnewsf对象的fid不为空，则为修改字段、转移字段接口。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findparenttidtbname | tidtbname | String |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| tbname | String | 数据表表名 |

**d.返回值示例与说明**

tidtbname

**e.功能实现说明**

判断

1. 判断tbname为空或为空字符串时，返回tidtbname值为error1
2. 根据tbname在模型数据表中查找该表的记录，去该表的parenttb，
3. 判断parenttb为空或为空字符串时，返回tidtbname值为error2，否者在根据parenttb在模型数据表中查找该父表的记录，通过父表的记录分别取到父表的tid和tbname，将其用“|”拼接，赋值给tidtbname，
4. 返回tidtbname；
   * 1. 修改字段顺序接口

**a.接口url:**

system/tableModel/updateFieldOrder.do

**b.接口说明:**

修改、批量修改字段顺序。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| updateFieldOrder | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| fid | String[] | 字段id的数组 |
| myorder | String[] | 字段显示顺序的数组 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "修改成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

①接收参数fid和myorder数组，遍历fid数组，得到fid[i]，结合对应myorder[i]生成sql语句

“UPDATE zjxx\_enewsf SET myorder = ‘myorder[i]’ WHERE fid = ‘fid[i]’”

②执行该sql更新数据库，返回result。

* + 1. 显示html代码接口

**a.接口url:**

system/tableModel/showHtml.do

**b.接口说明:**

利用freemarker读取模板html.txt文件，并根据前台传入的值替换模板中的值，生成html代码。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| showHtml | String | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| fform | String | 输入表单显示元素 |
| f | String | 字段名 |
| jsonstring | String | 选项json字符串，记录不同fform需要的不同选项 |
| fvalue | String | 字段初始值 |

**d.返回值示例与说明**

{

"html": "<input name=/"zjxxEnewsf.getF()/" type=/"text/" id=/"zjxxEnewsf.getF()/" value=/"<?=$ecmsfirstpost==1?/"/":ehtmlspecialchars(stripSlashes($r[zjxxEnewsf.getF()]))?>/" size=/"45/">/n <a onclick=/"window.open('ecmseditor/FileMain.php?type=2" //html代码

}

**e.功能实现说明**

①接收参数fform、f、jsonstring、fvalue。

②根据fform从html.txt文本中读取对应的默认html，例如fform=password，从模板html.txt中读取形如“<#elseif (fform)=="password">”的默认html。

③根据f、fform的值封装在Map中。

④当fform值为select、radio、checkbox时需要对fvalue进行解析（此时fvalue：value==key多个值之间用”\n”分割），取出每一组key、value。将其封装到Map中

~~⑤当fform值为dictionaries、dictionarylist时需要对解析jsonstring，得到其中的fformsize，通过fformsize的值（即为数据词典的fid）来调用数据词典controller中的getTitles（）接口，得到数据词典关联单选或多选的titles字符串（之间用“，”分割），解析titles字符串得到一组title作为单选或多选的值，再将其封装到Map中。~~

⑥当fform值为classselectinfo、roles、labels、people、institutions和dictionaries时，为关联字段，jsonstring中的fformsize即为关联栏目信息的classid，角色关联的父id，机构关联的父id，数据词典关联的父id，将其封装到Map中。（在后面的录入信息时，在添加信息页面将调用html，生成关联字段的button，点击其中的button，将分别调用要查找的关联信息接口，将返回值在新的弹出div中显示出来，以供添加信息时选择，选中那条信息，则返回id）。

⑥新添加了发布时间、所属机构、标题图片、标题四大基础系统字段的输出类型。（主要是为了解决系统字段被修改输出类型后，一样可以还原成系统字段的输出类型）

⑦解析jsonstring，jsonstring:{"fformsize":60,"fformwidth":"","fformheight":"","linkfieldtb":"","linkfieldshow":"","linkfieldval":"","samedata":"","editorys":""}其中jsonstring共包含8个字段，分别代表：表单元素长度；元素宽度，高度；关联显示值字段名；关联数据表名；关联显示字段；关联是否数据同步；编辑器样式；

可将其转化为json对象，获取其中对应的值，将需要用到的数据封装到Map中。

⑧利用freemark模板中的数据用Map中的值替换（例如Map中的键为key，将会自动替换模板中的${key}）。

⑨返回修改后的html。

* + 1. 查找系统模型信息接口

**a.接口url:**

system/tableModel/findModel.do

**b.接口说明:**

通过不同方式查找一个系统模型。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findModel | ZjxxModel  （说明见6.1.4） | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| mid | String | 系统模型id |
| mname | String | 系统模型名称 |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，1为成功，0为操作失败

"msg": "配置员工角色成功" //返回的信息；

}

“infos”：{

"canaddf": "表名:字段名, 表名:字段名, 表名:字段名", //可增加项

"caneditf": "表名:字段名, 表名:字段名, 表名:字段名", //可修改项

"cj": "用户名${account}密码${password}标题${a\_title}副标题${a\_ftitle}", //采集项

"definfovoteid": 0, //信息预设投票ID

"enter": "用户名${account}密码${password}标题${a\_title}副标题${a\_ftitle}特殊属性${a\_specialfield}发布时间${a\_newstime}标题图片${a\_titlepic}", //录入项

"isdefault": false, //是否默认模型

"listandf": "表名:字段名, 表名:字段名, 表名:字段名", //结合项

"listfile": "", //信息列表名称

"listtempvar": "表名:字段名, 表名:字段名, 表名:字段名", //列表模板项

"mid": "fc943551-a5d3-4603-9dcf-08d0506bb823", //模型ID

"mname": "t2的系统模型1", //模型名称

"mtemp": "[{/"fileds/":/"account,password,a\_title,a\_ftitle,a\_specialfield,a\_newstime,a\_titlepic/",/"htmls/":/"用户名:<input type='condlinkfield' name='account' value=''/><br>密码:<input type='password' name='password' value=''/><br>标题:<input type='text' name='a\_title' value=''/><br>副标题:<input type='text' name='a\_ftitle' value=''/><br>特殊属性:<input type='text' name='a\_specialfield' value=''/><br>发布时间:<input type='text' name='a\_newstime' value=''/><br>标题图片:<input type='text' name='a\_titlepic' value=''/>/"}]", //录入表单模板

"mustqenterf": "account,password,a\_title,a\_ftitle,a\_specialfield,a\_newstime,a\_titlepic,", //必填项

"myorder": 0, //排序

"mzs": "", //注释

"orderf": "account,password,a\_title,a\_ftitle,", //排序项

"printtempid": 0, //打印模板ID

"qenter": "用户名${account}密码${password}标题${a\_title}副标题${a\_ftitle}", //投稿项

"qmname": "t2的系统模型1", //模型别名

"searchvar": "account,password,a\_title,a\_ftitle,", //搜索项

"setandf": false, //结合相匹配

"showmod": true, //是否显示到前台导航

"tbname": "zjxx\_t2", //数据表名

"tempvar": "用户名${account}密码${password}标题${a\_title}副标题${a\_ftitle}", //内容模板项

"tid": "338055a8-15aa-4174-ac7b-0a65148125fb", //数据表ID

"uk": "account,password,a\_title,a\_ftitle,", //联合主键

"usemod": true //是否启用

}

}

**e.功能实现说明**

1. 检查参数mid、mname,若均为空、则返回“来源错误”
2. 若mid非空、则拼接加入查询条件、使用精确查询（=）
3. 若mname非空、则拼接加入查询条件、使用精确查询（=）
4. 根据组合的查询条件查询数据库、返回对应数据
   * 1. 根据classid查系统模型接口

**a.接口url:**

system/tableModel/findModelByClassid.do

**b.接口说明:**

该接口用于通过classid查找系统模型数据。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findModelByClassid | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |

**d.返回值示例与说明**

//信息

{

“infos”：{……}

}

**e.功能实现说明**

1. 接收参数classid，为空值则返回“来源错误”
2. 根据classid找到栏目信息中的系统模型mid。
3. 根据通过mid找到系统模型信息。
4. 返回info。
   * 1. 查找系统模型列表接口

**a.接口url:**

system/tableModel/findModelList.do

**b.接口说明:**

该接口用于查找系统模型列表。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findModelList | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| tbname | String | 数据表名 |
| page | String | 页码 |
| num | String | 每页显示信息数 |

**d.返回值示例与说明**

{

"page": { //分页信息

"page": 2, //当前页

"pages": 10, //总页数

"lines": 100, //信息总条数

"pageSize": "10", //每页显示信息数

"pageStart": 1 //起始页

},

"total": 1, //当前页信息数

"infos": [ //信息

{

"canaddf": "表名:字段名, 表名:字段名, 表名:字段名", //可增加项

"caneditf": "表名:字段名, 表名:字段名, 表名:字段名", //可修改项

"cj": "用户名${account}密码${password}标题${a\_title}副标题${a\_ftitle}", //采集项

"definfovoteid": 0, //信息预设投票ID

"enter": "用户名${account}密码${password}标题${a\_title}副标题${a\_ftitle}特殊属性${a\_specialfield}发布时间${a\_newstime}标题图片${a\_titlepic}", //录入项

"isdefault": false, //是否默认模型

"listandf": "表名:字段名, 表名:字段名, 表名:字段名", //结合项

"listfile": "", //信息列表名称

"listtempvar": "表名:字段名, 表名:字段名, 表名:字段名", //列表模板项

"mid": "fc943551-a5d3-4603-9dcf-08d0506bb823", //模型ID

"mname": "t2的系统模型1", //模型名称

"mtemp": "[{/"fileds/":/"account,password,a\_title,a\_ftitle,a\_specialfield,a\_newstime,a\_titlepic/",/"htmls/":/"用户名:<input type='condlinkfield' name='account' value=''/><br>密码:<input type='password' name='password' value=''/><br>标题:<input type='text' name='a\_title' value=''/><br>副标题:<input type='text' name='a\_ftitle' value=''/><br>特殊属性:<input type='text' name='a\_specialfield' value=''/><br>发布时间:<input type='text' name='a\_newstime' value=''/><br>标题图片:<input type='text' name='a\_titlepic' value=''/>/"}]", //录入表单模板

"mustqenterf": "account,password,a\_title,a\_ftitle,a\_specialfield,a\_newstime,a\_titlepic,", //必填项

"myorder": 0, //排序

"mzs": "", //注释

"orderf": "account,password,a\_title,a\_ftitle,", //排序项

"printtempid": 0, //打印模板ID

"qenter": "用户名${account}密码${password}标题${a\_title}副标题${a\_ftitle}", //投稿项

"qmname": "t2的系统模型1", //模型别名

"searchvar": "account,password,a\_title,a\_ftitle,", //搜索项

"setandf": false, //结合相匹配

"showmod": true, //是否显示到前台导航

"tbname": "zjxx\_t2", //数据表名

"tempvar": "用户名${account}密码${password}标题${a\_title}副标题${a\_ftitle}", //内容模板项

"tid": "338055a8-15aa-4174-ac7b-0a65148125fb", //数据表ID

"uk": "account,password,a\_title,a\_ftitle,", //联合主键

"usemod": true //是否启用

}

]

}

**e.功能实现说明**

1. 检查page参数、为空则设置默认值1，检查num参数、为空则设置默认值30。
2. 检查tbname参数，若非空则根据tbname的值拼接查询条件、使用精确查询（=）
3. 根据page和num生成页码条件（limit）
4. 根据生成的sql语句查询数据，返回数据。
   * 1. 删除系统模型接口

**a.接口url:**

system/tableModel/deleteModel.do

**b.接口说明:**

该接口用于删除系统模型

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findModel | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| mid | String | 系统模型id |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "删除成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

①传入参数mid，在栏目表查询是否有栏目绑定此系统模型，如果有，提示不能删除。

②如果没有，执行SQL语句“DELETE FROM zjxx\_model WHERE mid = ‘mid’;”删除对应mid的系统模型，返回result。

* + 1. 增加/修改系统模型接口

**a.接口url:**

system/tableModel/addOrUpdateModel.do

**b.接口说明:**

（1）如果传入参数zjxxModel对象的mid为空，则该接口用于增加、复制系统模型。

（2）如果传入参数zjxxModel对象的mid不为空，则该接口用于修改系统模型。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| addOrUpdateModel | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| zjxxModel | ZjxxModel | 系统模型对象 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "增加成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

（1）如果传入参数zjxxModel对象的mid为空：

①接收参数zjxxModel对象，生成UUID并赋值给zjxxModel对象的mid属性。

②将zjxxModel对象插入zjxx\_model表中，返回result。

（2）如果传入参数zjxxModel对象的mid不为空：

①接收参数zjxxModel对象，根据zjxxModel对象的属性更新zjxx\_model表中的数据，返回result。

* + 1. 查询分表列表接口

**a.接口url:**

system/tableModel/findPartTableList.do

**b.接口说明:**

该接口用于通过一个主表的tbname查询所有分表列表。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findPartTableList | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| tbname | String | 数据表名 |

**d.返回值示例与说明**

{

"infos": [ //信息

{

"ismaster": false, //是否为主表

"masterb": "zjxx\_t2", //主表名

"tbname": "zjxx\_t2\_1", //表名

"tid": "b7683f90-5bff-42e0-b063-7002eb1af66e" //表ID

},

{…}

]

}

**e.功能实现说明**

①检查参数tbname，若为空则返回信息“来源错误”

②非空则根据tbname查找信息。若是主表、则直接查询所有分表信息。若是分表、则找到对应主表后在查询所有分表信息

③最终返回主表于其全部分表的信息。

* + 1. 增加分表接口

**a.接口url:**

system/tableModel/addPartTable.do

**b.接口说明:**

该接口用于增加分表。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| addPartTable | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| masterb | String | 主表名称 |
| tbname | String | 数据表名 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "增加成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

①接收参数masterb和tbname，生成一个ZjxxEnewstable对象zjxxEnewstable。

②生成UUID，将UUID、masterb、tbname赋值给zjxxEnewstable对象，设置zjxxEnewstable对象的ismaster属性为false。

③将zjxxEnewstable对象插入zjxx\_enewstable数据表。

④在数据库建立名为tbname、tbname\_gd、tbname\_wk的三张表。

⑤通过masterb找到该分表的主表数据，通过主表数据得到主表的分表集合datatbs，将该tbname拼接到datatbs中。

⑥更新主表数据，返回result。

* + 1. 删除分表接口

**a.接口url:**

system/tableModel/deletePartTable.do

**b.接口说明:**

该接口用于删除一个分表。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| deletePartTable | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| tid | String | 分表id |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "删除成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

①接收参数tid，通过tid找到符合的分表对象zjxxEnewstable，得到zjxxEnewstable对象的属性tbname和masterb。

②执行SQL语句“DELETE FROM zjxx\_enewstable WHERE tid = ‘tid’;”删除zjxx\_enewstable表中符合该tid的数据。

③执行SQL语句“DROP TABLE tbname; DROP TABLE tbname\_gd; DROP TABLE tbname\_wk;”删除数据库中的tbname表和其归档表、微客表。

④通过masterb找到该分表的主表，得到主表的分表集合字段datatbs，使用replace(tbname+”,”,””)修改datatbs，更新主表数据，返回result。

* + 1. 设置默认系统模型接口

**a.接口url:**

pub/admin/system/tableModel/setDefaultModel.do

**b.接口说明：**

设置当前系统模型为默认系统模型，其他系统模型为非默认。

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| setDefaultModel | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| mid | String | | 系统模型id | |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "增加成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

1. 接收参数mid，通过mid找到系统模型model。
2. 通过model获得tbname。
3. 设置属于该tbname的所有系统模型为非默认。（isdefault为0）。设置属于该tbname且mid=mid的系统模型为默认。（isdefault为1）
4. 返回result。
   * 1. 查询修改信息数据接口

**a.接口url:**

pub/admin/information/Infohtml/findUpdateInfo.do

**b.接口说明:**

该接口用于后台修改信息数据前，查询对应栏目下的要修改的信息数据。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findUpdateInfo | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| id | String | 信息id |

**d.返回值示例与说明**

//信息

{

“infos”：{……}

}

**e.功能实现说明**

1. 接收参数id、classid，有任意一个空值则返回“来源错误”
2. 根据classid找到栏目信息中的系统模型mid
3. 根据通过mid找到系统模型信息中的表名（tbname）以及录入项（enter）
4. 分解enter重新组合成字段拼接字符串：字段1,字段2,字段3
5. 根据id和tbname查询对应表中的enter所示字段以及系统字段，返回数据
   * 1. 查询信息数据接口

**a.接口url:**

pub/admin/information/Infohtml/findInfo.do

**b.接口说明:**

该接口用于查询对应栏目下的信息数据。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findInfo | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| id | String | 信息id |
| method | String | method为”modify”时为修改信息查找，method为“show”时为显示信息时查找信息 |

**d.返回值示例与说明**

//信息

{

“infos”：{……}

}

**e.功能实现说明**

1. 接收参数id、method、classid，有任意一个空值则返回“来源错误”
2. 根据classid找到栏目信息中的系统模型mid
3. 判断method为“modify”还是为“show”

1.当method为“modify”时

1. 根据通过mid找到系统模型信息中的表名（tbname）以及录入项（enter）
2. 分解enter重新组合成字段拼接字符串：字段1,字段2,字段3
3. 根据id和tbname查询对应表中的enter所示字段以及系统字段，返回 数据

2.当method为“show”时

1）根据通过mid找到系统模型信息中的表名（tbname）以及内容项（tempvar）

2）分解tempvar重新组合成字段拼接字符串：字段1,字段2,字段3

3）根据id和tbname查询对应表中的tempvar所示字段，返回数据

* + 1. 查询分表记录数接口

**a.接口url:**

system/tableModel/findCountByTbname.do

**b.接口说明:**

该接口用于查询分表记录数。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findCountByTbname | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| tbname | String | 数据表名 |

**d.返回值示例与说明**

{

result:{flag:1,msg:查询成功},

total:10

}

**e.功能实现说明**

①检查参数tbname，若为空则返回信息“来源错误”

②非空则根据tbname查找对应表的记录数。

③返回info。

* + 1. 查询信息列表接口

**a.接口url:**

pub/admin/information/Infohtml/findInfoList.do

**b.接口说明:**

该接口用于查询对应栏目下的信息数据列表。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findInfoList | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| userid | String | 用户id |
| page | String | 当前页码 |
| num | String | 每页显示信息条数 |
| orderType | String | 排序类型 |
| tablechange | String | gd表示查询归档表 |
| searchAttr | String | 搜索方法的属性，如：  1为置顶、2为推荐、3为头条、4为未审核、5为已审核、6为投稿、7为签发、8为我的信息 |
| searchValue | String[] | 搜索方法的关键字数组 |
| searchField | String[] | 搜索方法的字段数组，即该表包含的字段，如：  标题、机构、副标题等。 |
| searchTime | String | 搜索方法的时间，如：  1天、2天、一周、1个月、3个月、6个月、1年。 |

**d.返回值示例与说明**

{

"page": { //分页信息

"page": 2, //当前页

"pages": 10, //总页数

"lines": 100, //信息总条数

"pageSize": "10", //每页显示信息数

"pageStart": 1 //起始页

},

"total": 1, //当前页信息数

"infos": [ //信息

{

……

},

{

……

}

]

}

**e.功能实现说明**

1. 检查参数classid、为空则返回信息“来源错误”。非空则通过classid找到栏目对象，通过栏目对象找到mid，通过mid找到系统模型，通过系统模型找到tbname。
2. 根据tbname找到相关分表
3. 如果tablechange非空，则根据tablechange修改各表名至对应归档表
4. 一次检测searchAttr、searchValue、searchField、searchTime参数，非空值、则根据对应数值拼接sql查询条件
5. 若orderType非空则拼接order by排序内容
6. 根据num和page拼接limit内容。若为空值则使用默认值
7. 执行生成好的sql语句、查询并返回数据
   * 1. 删除信息接口

**a.接口url:**

pub/admin/information/Infohtml/deleteInfo.do

**b.接口说明:**

该接口用于删除、批量删除某个栏目下的信息

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| deleteInfo | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| ids | String[] | 信息id数组 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "删除成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

①传入参数classid、ids，通过classid找到栏目对象，通过栏目对象找到mid，通过mid找到系统模型，通过系统模型找到tbname。

②遍历ids数组，将表名于id传入**同步删除方法**

③返回result

* + 1. 操作信息接口

**a.接口url:**

pub/admin/information/Infohtml/operateInfo.do

**b.接口说明:**

该接口用于添加/修改信息内容。

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| operateInfo | |  | | Result |
|  | |  | |  |
| infoJson | String | | 信息内容组成的json，因为不同栏目信息字段不同，所以动态生成json，格式为  {  "id":"",//信息id  "classid":"",//栏目id  "recommendation":"0",//推荐  "headline":"0"， //头条  "top":"0"， //置顶  "time":"0"， //修改信息的发布时间  "check":"0" //是否审核  ... //其他系统模型字段    } | |
| userid | String | | 用户ID | |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "修改成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

①直接调用[**信息操作的方法**](#_5.1.9__信息操作方法) **5.1.9**，传入infojson

②根据方法返回值，返回信息

* + 1. 获取不同基础表字段接口

**a.接口url:**

system/tableModel/getZjxxBaseTableField.do

**b.接口说明:**

通过基础表的tbname获取该表的字段。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| getZjxxBaseTableField | ZjxxBaseTableField  （说明见6.1.5） | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| baseTbname | String[] | 基础表的表名数组 |

**d.返回值示例与说明**

{

“infos”：[ {

"columnName": "id", //字段名

"dataType": "varchar", //字段类型

"characterMaxImumLength": "48", //字段长度

"columnComment": "姓名", //字段备注

"isNullable": "no", //是否为空

"columnKey": "PRI" //是否为主键

},

{}

]

}

**e.功能实现说明**

①接收参数baseTbname数组，每次遍历该数组得到一个基础表名'tbname'，依次执行SQL语句

select column\_name,data\_type,character\_maxImum\_length,column\_comment,is\_nullable,column\_key

from information\_schema.COLUMNS

where TABLE\_SCHEMA='javaplat' and TABLE\_NAME='tbname'

②即可得到该表的所有字段信息，返回ZjxxBaseTableField类。

* + 1. 字段数据转移接口

**a.接口url:**

/pub/admin/system/tableModel/transferFieldData.do

**b.接口说明:**

字段修改所属表时数据转移用

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| transferFieldData | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| formTbname | String | 转出数据表名 |
| toTbname | String | 转入数据表名 |
| fname | String | 转移的字段名 |
| start | Int | 转移的起点下标 |
| size | Int | 转移多少条数据 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "200", //成功转移数量

"flag": "1" //标识

}

}

**e.功能实现说明**

1. 检查参数、formTbname、toTbname、fname，有任意一项为空值则返回信息 “来源错误”
2. 检查参数start、size、若为空则设置默认值start=0、size=200
3. 根据formTbname、fname编写查询语句、根据start、size控制limit
4. 遍历查询结果、依次执行update语句修改toTbname中的fname值。并记录转移成功数
5. 最后将成功转移成功数返回。
   * 1. 栏目（其他）转换信息接口

**a.接口url:**

pub/admin/column/manageColumn/transferToInfo.do

**b.接口说明:**

将栏目或其他数据表（八大类）转化为信息表数据。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| transferToInfo | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 目标栏目id |
| id | String | 需要转换的信息id |
| formType | int | 信息来源类别：   1. 栏目表 2. 机构表 3. 数据词典表 4. 角色表 5. 人员表 6. 标签表 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "转换成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

1. 直接将3个参数传入 **栏目（其他）转信息方法** [5.1.10](#_5.1.10__其他转信息方法)
2. 返回方法传出的返回值
   * 1. 信息转栏目(其他)接口

**a.接口url:**

pub/admin/ column/manageColumn/infoTransferToOthers.do

**b.接口说明:**

将信息数据转换为栏目&其他表数据

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| infoTransferToOthers | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| fid | String | 父项id |
| Id | String | 信息id |
| Tbname | String | 信息表名 |
| ctype | Int | 转换目标类型：  1.栏目表  2.机构表  3.数据词典表  4.角色表  5.人员表  6.标签表 |
| isshow | int | 是否隐藏（仅限栏目生效） 0隐藏 1显示 |
| Modid | String | 系统模型id（仅限栏目生效） |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "转换成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

1. 直接将参数传入 **信息转栏目（其他）方法** [5.1.11](#_5.1.11__信息转其他方法)
2. 返回方法传出的返回值
   * 1. 批量信息归档接口

**a.接口url:**

pub/admin/information/Infohtml/batchInfoArchive.do

**b.接口说明:**

栏目信息归档/批量选择信息归档

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| batchInfoArchive | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| ids | String | 转入数据表名 |
| start | Int | 转移的起点下标 |
| size | Int | 转移多少条数据 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "200", //成功归档数量

"flag": "1" //标识

}

}

**e.功能实现说明**

1. 检查参数、classid，为空值则返回信息 “来源错误”
2. 检查参数start、size、若为空则设置默认值start=0、size=200
3. 调用[归档方法](#_5.1.8__归档方法)传入classid、ids、start、size
4. 把归档方法的返回值存入result归档数量、返回数据。
   * 1. 批量信息修改接口

**a.接口url:**

pub/admin/information/Infohtml/batchUpdateInfo.do

**b.接口说明:**

批量修改信息单个字段（批量审核、批量置顶等操作）

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| batchUpdateInfo | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| ids | String | Id集合、使用逗号分隔：  Id1,id2,id3, |
| Classid | String | 栏目id |
| key | String | 需要修改的字段名称 |
| value | Int | 需要修改的字段值 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "批量修改成功", //成功转移数量

"flag": "1" //标识

}

}

**e.功能实现说明**

1. 检查参数、ids、key、任意值为空值则返回信息 “来源错误”
2. 拆分ids后组成sql条件【where id in (‘id1’, ‘id2’, ‘id3’)】
3. 根据key和value参数组成update语句、并接上id条件
4. 执行sql语句
5. 返回result
   * 1. 信息转移/复制接口

**a.接口url:**

pub/admin/information/Infohtml/batchCopyOrTransferInfo.do

**b.接口说明:**

信息批量转移/复制

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| batchCopyOrTransferInfo | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| dotype | Int | 操作类型 0代表移动 1代表复制 |
| toClassid | String | 目标栏目id |
| classid | String | 栏目id |
| ids | String | 转移Id集合 |
| start | Int | 转移的起点下标 |
| size | Int | 转移多少条数据 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "200", //成功转移/复制数量

"flag": "1" //标识

}

}

**e.功能实现说明**

1. 检查参数、classid，toClassid，为空值则返回信息 “来源错误”
2. 检查参数start、size、若为空则设置默认值start=0、size=200
3. 根据classid，toclassid分别获取对应系统模型信息，再进一步获取到数据表以及分表信息。
4. 若2个classid的数据表不同，则返回提示“数据表不符”
5. 检查参数ids、若非空、则拆分后组成sql条件【where id in (‘id1’, ‘id2’, ‘id3’)】
6. 根据操作类型判断，若是转移操作，则由sql条件生成update语句，直接修改classid并返回
7. 若是赋值操作，则根据classid结合id条件 组成sql查询语句执行
8. 遍历查询结果、依次把数据内容复制进目标栏目（注意主表、分表信息）。并记录操作信息数
9. 最后将记录操作信息数返回。
   * 1. 内容页html生成接口

**a.接口url:**

pub/admin/information/Infohtml/createContentHtml.do

**b.接口说明:**

用于生成内容页html

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| createContentHtml | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 信息id |
| classid | String | 栏目id |
| filename | String | 文件名 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "操作成功", //提示信息

"flag": "1" //标识

}

}

**e.功能实现说明**

1. 检查参数、id、classid、filename，皆为空值则返回信息 “来源错误”，filename为空且id、classid中有任意一个空值同样返回信息“来源错误”
2. 若id、classid任意一值为空，分解filename、取出第一位字符找到对应缓存信息表、再根据缓存信息表和filename找到classid和id。
3. 如果classid和id不为空，将classid和infoid传入[生成文件名方法](#_5.1.1__生成文件名方法)，生成文件名。
4. 把classid、id、filename、缓存信息表分表cacheInfoTbname传入[**生成页面信息方法**](#_5.1.2__生成页面信息方法)
5. 返回result
   * 1. 列表页html生成接口

**a.接口url:**

pub/admin/information/Infohtml/createListHtml.do

**b.接口说明:**

用于生成列表页html

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| createListHtml | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| filename | String | 文件名 |
| page | int | 指定页码 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "操作完成", //信息提示

"flag": "1" //标识

}

}

**e.功能实现说明**

1. 检查参数、classid、filename，都为空值则返回信息 “来源错误”
2. classid为空时、分解filename、取出第一位字符找到对应缓存信息表，再根据缓存信息表和filename找到classid。
3. 如果classid不为空，将classid传入[生成文件名方法](#_5.1.1__生成文件名方法)，生成文件名。
4. 把classid、page（为空默认查第一页）、filename、缓存信息表分表cacheInfoTbname传入[**生成列表信息方法**](#_5.1.3__生成列表信息方法)
5. 返回result
   * 1. 条件列表页html生成接口

**a.接口url:**

pub/admin/information/Infohtml/createConditionListHtml.do

**b.接口说明:**

用于生成列表页html

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| createConditionListHtml | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| filename | String | 文件名 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "操作完成", //信息提示

"flag": "1" //标识

}

}

**e.功能实现说明**

1. 检查参数、classid、filename，都为空值则返回信息 “来源错误”
2. classid为空时、分解filename、取出第一位字符找到对应缓存信息表、再根据第二位开始的字符串找到对应信息的classid
3. 根据classid查询栏目信息、生成栏目对象

\*\*\*\*\*\*

* + 1. 威客列表查询接口

**a.接口url:**

pub/admin/information/Infohtml/findModifyRecordList.do

**b.接口说明:**

查询威客信息列表

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findModifyRecordList |  | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 信息id |
| classid | String | 栏目id |
| editer | String | 修改人 |
| title | String | 标题 |
| stime | Int | 查询时间范围起点 |
| etime | Int | 查询时间范围终点 |
| page | int | 页码 |
| pagesize | int | 每页信息数 |

**d.返回值示例与说明**

{

"page": { //分页信息

"page": 2, //当前页

"pages": 10, //总页数

"lines": 100, //信息总条数

"pageSize": "10", //每页显示信息数

"pageStart": 1 //起始页

},

"total": 1, //当前页信息数

"infos": [ //信息

{

……

},

{

……

}

]

}

**e.功能实现说明**

1. 检查参数、classid、id，任意值为空值则返回信息 “来源错误”
2. 根据classid查询信息所在系统模型id以及内容模板id、再根据系统模型id取得数据表名并转化为威客表名
3. 若参数id非空、则拼接条件、精确查询信息id（=） \*注意信息id≠威客id
4. 若参数editer非空、则拼接条件、精确查询修改人（=）
5. 若参数title非空、则拼接条件、模糊查询（like）
6. 若参数stime非空、则拼接条件、比较查询修改时间（>）
7. 若参数editer非空、则拼接条件、比较查询修改时间（<）
8. 根据page和pagesize组合limit条件。若为空则设置默认值
9. 执行查询语句、返回数据
   * 1. 威客信息查询接口
10. **接口url:**

pub/admin/information/Infohtml/findModifyRecordById.do

**b.接口说明:**

信息批量转移/复制

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findModifyRecordById | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 威客id |
| tbname | String | 目标数据表名 |
| classid | String | 栏目id |

**d.返回值示例与说明**

{

"infos": [ //信息

{

……

},

{

……

}

]

}

**e.功能实现说明**

1. 检查参数、id，为空值则返回信息 “来源错误”
2. 检查参数、tbname、classid，皆为空值则返回信息 “来源错误”
3. 若tbname为空。根据classid查询信息所在系统模型id以及内容模板id、再根据系统模型id取得数据表名并转化为威客表名
4. 若tbname非空。则直接转化为威客表名
5. 根据表名和id查询数据并返回
   * 1. 信息转换栏目接口

**a.接口url:**

Transferinfo.do

**b.接口说明:**

将信息数据转换为栏目

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| Transferinfo | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| id | String | | 信息的id\ | | |
| title | String | | 信息的名字 | | |
| tbname | String | | 转换目标的表名(栏目的主表名称) | | |
| fid | String | | 选择父栏目id | | |
| isshow | String | | 是否显示栏目（1显示，0不显示） | | |
| btype | String | | 传入“栏目” | | |
| modid | String | | 信息转换栏目时绑定的系统模型 | | |
| origintable | String | | 来源表名字段 | | |
| Targettable | String | | 目标表名字段（多个表名以“，”隔开） | | |
| classid | String | | 信息所属的栏目id | | |

**d.返回值示例与说明**

{

"flag": "1", //返回值标志

"msg": "转移成功", //返回值信息

}

**e.功能实现说明**

1. 接收参数id、title、isshow，mid判断有任意一个空值则返回“来源错误”。
2. 根据tbname、id、fid获取来源数据，若找不到数据则同样返回“来源错误”。
3. 根据传入参数mid判断如果为空，则通过fid查找所属栏目本身绑定的品论的系统 模型，将其赋予modid作为字段值，拼接sql语句。
4. 如果传入参数mid不为空，则直接赋值给modid拼接sql语句。
5. 根据id的值和通过classid获取的表名获取targettable的值，然后更新这 个值（把原有的值附加新的表名用逗号隔开）。
6. 执行sql语句。
7. 再次拼接sql，这句sql值只需要获取id的值作为classid，增加一条记录到栏目所 有副表（副表只需要这条id的记录存在，其他字段值可以为空）。
8. 返回result。
   * 1. 信息转换其他接口

**a.接口url:**

Transferinfo.do

**b.接口说明:**

将信息数据转换为栏目

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| InfoTransfer | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| Id | String | | 信息id | | |
| title | String | | 信息的名字 | | |
| tbname | String | | 转换目标的表名(栏目的主表名称) | | |
| btype | String | | 转换的类型 | | |
| origintable | String | | 来源表名字段 | | |
| Targettable | String | | 目标表名字段（多个表名以“，”隔开） | | |
| classid | String | | 信息所属的栏目id | | |

**d.返回值示例与说明**

{

"flag": "1", //返回值标志

"msg": "转移成功", //返回值信息

}

**e.功能实现说明**

**调用信息转换成其他的方法5.1.11**

* + 1. 信息转换人员接口

**a.接口url:**

infoTransfer.do

**b.接口说明:**

将信息数据转换为栏目

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| Transferinfo | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| id | String | | 信息的id | | |
| title | String | | 信息的名字 | | |
| tbname | String | | 转换目标的表名 | | |
| orgid | String | | 机构的id | | |
| btype | string | | 传入“人员” | | |
| origintable | String | | 来源表名字段 | | |
| Targettable | String | | 目标表名字段（多个表名以“，”隔开） | | |
| classid | String | | 信息所属的栏目id | | |

**d.返回值示例与说明**

{

"flag": "1", //返回值标志

"msg": "转移成功", //返回值信息

}

**e.功能实现说明**

1. 接收参数id、titl、orgid，判断有任意一个空值则返回“来源错误”。
2. 根据接收的参数拼接sql，向人员表中添加一条数据。
3. 再次拼接sql，向机构员工角色表中添加数据。例如;insert into 表名 (id,oid,rids,peid) values(‘’,’’,’’,’’),其中的oid就是参数orgid的值，peid即为参数id得值。
4. 根据id的值和通过classid获取的表名获取targettable的值，然后更新这个值（把原有的值附加新的表名用逗号隔开）。
5. 如果再次转换成其他机构下的人员，更新机构人员表的oid字段中间以，隔开。
6. 执行sql，往信息表里插入一条记录。
7. 返回result。
   * 1. 增加阅读记录接口

**a.接口url:**

pub/admin/information/Infohtml/addReadingLog.do

**b.接口说明:**

（1）必须传入classid，infoid，peopleid3个参数。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| addReadingLog | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| infoid | String | 信息id |
| peopleid | String | 人员id |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "成功", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

（1）根据classid，infoid，peopleid查询有无记录。

（2）若有，则证明已该人员已阅读过此条信息，返回已读过这条信息。

（3）若无，则插入一条记录，返回已阅读。

* + 1. 获取未读条数接口

**a.接口url:**

pub/admin/information/Infohtml/getNoReadCount.do

**b.接口说明:**

（1）传classid和peopleid，查询该人员指定栏目下有多少条未读消息,以及未读消息的id集合

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| getNoReadCount | Infor | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| peopleid | String | 人员id |

**d.返回值示例与说明**

{  
    "infos":[  
        {  
            "infoid":"1e1546fe-e1d2-4b23-9462-49f819b90abe"  
        },  
        {  
            "infoid":"5a825b38-bf4d-42e1-95b2-38d6edcc23f3"  
        },  
        {  
            "infoid":"8885f1d0-cb92-4178-b0ca-9604ffd3bf2d"  
        },  
        {  
            "infoid":"dd4d9c6b-4ce5-4584-aab0-6d9a347c3e20"  
        }  
    ],  
    "page":null,  
    "result":{  
        "flag":1,  
        "msg":"success",  
        "orgid":"",  
        "rid":"",  
        "token":"",  
        "ttime":"",  
        "userId":""  
    },  
    "total":4  
}

**e.功能实现说明**

（1）根据classid查询该栏目下有多少信息，把所有的id集合存放在list中。

（2）根据classid和peopleid查询该人员阅读过该栏目下的多少信息，把阅读过的信息id存放在另一list中

（3）根据2个list统计总共有多少条未读信息，并把未读信息的id返回。

* + 1. 查询阅读记录接口

**a.接口url:**

pub/admin/information/Infohtml/findReadingLogList.do

**b.接口说明:**

（1）单独传id或者传classid、infoid、peopleid三个参数，查询指定人的指定记录。

（2）传classid，查询指定栏目所有人的所有记录。

（3）传classid和infoid，查询指定信息所有人的记录。

（4）传peopleid，查询指定人员所有记录。

（5）传peopleid和classid，查询指定人的指定栏目的所有记录。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findReadingLogList | Infor | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 阅读记录id |
| classid | String | 栏目id |
| infoid | String | 信息id |
| peopleid | String | 人员id |

**d.返回值示例与说明**

{  
    "infos":[  
        {  
            "classid":"be68990e-6985-4b7e-9620-6d9c31c9e683",  
            "infoid":"64585518-23c5-4e85-89c6-0e08fa4cbcbd",  
            "peopleid":"9db7c86d-a6c2-4bd4-ad02-3dcc325f1b40",  
            "id":"f90a8006-605b-488d-ab9c-e425c4805184",  
            "readtime":1495093555  
        }  
    ],  
    "page":null,  
    "result":{  
        "flag":1,  
        "msg":"success",  
        "orgid":"",  
        "rid":"",  
        "token":"",  
        "ttime":"",  
        "userId":""  
    },  
    "total":1  
}

**e.功能实现说明**

（1）判断传入的参数是否存在。

（2）根据传入的参数查询阅读记录表

（3）返回查询到的阅读记录。

* + 1. 查询阅读记录接口

**a.接口url:**

pub/admin/information/Infohtml/findReadingLogList.do

**b.接口说明:**

（1）单独传id或者传classid、infoid、peopleid三个参数，查询指定人的指定记录。

（2）传classid，查询指定栏目所有人的所有记录。

（3）传classid和infoid，查询指定信息所有人的记录。

（4）传peopleid，查询指定人员所有记录。

（5）传peopleid和classid，查询指定人的指定栏目的所有记录。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findReadingLogList | Infor | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 阅读记录id |
| classid | String | 栏目id |
| infoid | String | 信息id |
| peopleid | String | 人员id |

**d.返回值示例与说明**

{  
    "infos":[  
        {  
            "classid":"be68990e-6985-4b7e-9620-6d9c31c9e683",  
            "infoid":"64585518-23c5-4e85-89c6-0e08fa4cbcbd",  
            "peopleid":"9db7c86d-a6c2-4bd4-ad02-3dcc325f1b40",  
            "id":"f90a8006-605b-488d-ab9c-e425c4805184",  
            "readtime":1495093555  
        }  
    ],  
    "page":null,  
    "result":{  
        "flag":1,  
        "msg":"success",  
        "orgid":"",  
        "rid":"",  
        "token":"",  
        "ttime":"",  
        "userId":""  
    },  
    "total":1  
}

**e.功能实现说明**

（1）判断传入的参数是否存在。

（2）根据传入的参数查询阅读记录表

（3）返回查询到的阅读记录。

* + 1. 获取未读条数接口

**a.接口url:**

pub/admin/information/Infohtml/getNoReadCount.do

**b.接口说明:**

（1）传classid和peopleid，查询该人员指定栏目下有多少条未读消息,以及未读消息的id集合

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| getNoReadCount | Infor | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| peopleid | String | 人员id |

**d.返回值示例与说明**

{  
    "infos":[  
        {  
            "infoid":"1e1546fe-e1d2-4b23-9462-49f819b90abe"  
        },  
        {  
            "infoid":"5a825b38-bf4d-42e1-95b2-38d6edcc23f3"  
        },  
        {  
            "infoid":"8885f1d0-cb92-4178-b0ca-9604ffd3bf2d"  
        },  
        {  
            "infoid":"dd4d9c6b-4ce5-4584-aab0-6d9a347c3e20"  
        }  
    ],  
    "page":null,  
    "result":{  
        "flag":1,  
        "msg":"success",  
        "orgid":"",  
        "rid":"",  
        "token":"",  
        "ttime":"",  
        "userId":""  
    },  
    "total":4  
}

**e.功能实现说明**

（1）根据classid查询该栏目下有多少信息，把所有的id集合存放在list中。

（2）根据classid和peopleid查询该人员阅读过该栏目下的多少信息，把阅读过的信息id存放在另一list中

（3）根据2个list统计总共有多少条未读信息，并把未读信息的id返回。

* + 1. 增加点击量接口

**a.接口url:**

pub/admin/information/Infohtml/clickAddOne.do

**b.接口说明:**

（1）传classid和id，增加信息的点击量。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| clickAddOne | Result | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| id | String | 信息id |

**d.返回值示例与说明**

{  
    "flag":1,  
    "msg":"成功"  
}

**e.功能实现说明**

（1）根据classid和id查询该条信息。

（2）将信息的onclick字段+1 ，更新信息。

* + 1. 获取信息点击量接口

**a.接口url:**

pub/admin/information/Infohtml/getClickCount.do

**b.接口说明:**

（1）获取信息的点击量。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| getClickCount | Infor | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| id | String | 信息id |

**d.返回值示例与说明**

{  
    "infos":[  
        {  
            "onclick":1  
        }  
    ],  
    "page":null,  
    "result":{  
        "flag":1,  
        "msg":"查询成功",  
          },  
    "total":null  
}

**e.功能实现说明**

（1）根据classid和id查询该条信息。

（2）将信息的onclick字段返回。

* 1. 工作流模块
     1. 增加工作流模型接口

**a.接口url:**

pub/admin/system/workflow/ addWorkflow.do

**b.接口说明：**

创建一个工作流模型

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| addWorkflow | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| wfname | String | | 工作流名称 | |
| wftext | String | | 工作流描述 | |
| sysModelId | String | | 系统模型Id | |
| myorder | String | | 顺序 | |
| orgId | String | | 机构Id | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "添加模型成功" //返回的信息；

}

}

**e.功能实现说明**

1. 产生一个Activiti的model对象。
2. 把wfname设置到对象name中，wfname,wftest, sysModelId,myorder，orgId以json字符串的形式存到对象中的MetaInfo变量中，执行repositoryService接口的saveModel方法，传入model对象参数，就把model对象存入数据库
3. 创建一个json对象editorNode，editorNode对象放id，resourceId和stencilset这3个变量，id的值为canvas，resourceId的值为canvas，stencilset是一个json对象，里面放namespace变量，值为http://b3mn.org/stencilset/bpmn2.0#。在执行repositoryService接口的addModelEditorSource方法，传入Model对象的id和editorNode对象的字符化两个参数。
4. 创建Resultd对象，设置msg为添加模型成功，flag为0，并返回这个Result对象。
   * 1. 修改工作流模型

**a.接口url:**

pub/admin/system/workflow/updateWorkflow.do

**b.接口说明：**

修改已经创建的工作流模型

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| updateWorkflow | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| wfname | String | | 工作流名称 | |
| wftext | String | | 工作流描述 | |
| sysModelId | String | | 系统模型Id | |
| myorder | String | | 顺序 | |
| modelId | String | | 工作流模型Id | |
| orgId | String | | 机构Id | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "修改模型成功" //返回的信息；

}

}

**e.功能实现说明**

1. 把参数wftext，sysModelId，myorder，orgId放到一个json对象中。在把这个json对象字符化赋值到MetaInfo变量中。
2. 把wfname，modelId，MetaInfo这四个变量放到一个map中。
3. 执行对应dao层的接口的修改方法，参入map参数。
4. 创建Resultd对象，设置msg为修改模型成功，flag为0，并返回这个Result对象。
   * 1. 修改工作流模型查找信息接口

**a.接口url:**

pub/admin/system/workflow/searchWorkflowinfo.do

**b.接口说明：**

修改工作流模型前查找该模型的数据

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| searchWorkflowinfo | | Map<String,Object> | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| modelId | String | | 工作流模型Id | |

**d.返回值示例与说明**

{

" sysModelId ": "HI9385938 ", //配置工作流模型Id

"description": "aa", //工作流描述

"wfname": "aaaa", //工作流名称

"myorder": "34", //顺序

“orgId”：“” //机构Id

}

**e.功能实现说明**

1. 根据modelId查到这个model对象。
2. 把model中的name,description, sysModelId,myorder，orgId数据放到map中，其中description, sysModelId,myorder，orgId这三个数据存在model中的MetaInfo字段中。
3. 返回这个map。
   * 1. 查找工作流模型列表接口

**a.接口url:**

pub/admin/system/workflow/searchWorkflow.do

**b.接口说明：**

查询工作流模型列表，如果传modelName或orgId参数，用这个orgId 或modelName来模糊查找，如果不传，是对所有模型的搜索。

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| searchWorkflow | | infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| modelName | String | | 用模型名作为条件搜索 | |
| orgId | String | | 机构Id | |
| pageSize | Integer | | 每页显示的尺寸 | |
| nowPage | Integer | | 当前页 | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [

{

"sysModelId ": "837423", //系统模型Id

"modelid": "15001", //模型的id

"addtime": "Fri Aug 05 21:15:06 CST 2016",//模型增加的时间

"wfname": "aaaa", //工作流模型的名称

“orgId”：“” //机构Id

},

{}

]

}

**e.功能实现说明**

1. 根据参数拼接sql语句，把得到的list数据放到infor对象中的infos字段。把得到数据的长度放到infor对象中的total字段中。
2. 根据参数拼接sql语句，把得到的counts数据放到infor对象中的page对象的lines变量中，在给page对象设置page,pages,pageSize,pageStart字段。
3. 最后返回infor对象。
   * 1. 删除工作流模型

**a.接口url:**

pub/admin/system/workflow/deleteWorkflow.do

**b.接口说明：**

把已经存在的工作流模型删除掉

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteWorkflow | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| modelId | String | | 模型Id | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "删除模型成功" //返回的信息；

}

}

**e.功能实现说明**

1. 执行repositoryService接口的deleteMode方法，传入modelId参数，就可以删除这个模型。
2. 创建Resultd对象，设置msg为删除模型成功，flag为0，并返回这个Result对象。
   * 1. 部署工作流模型接口

**a.接口url:**

pub/admin/system/workflow/deploy.do

**b.接口说明：**

部署工作流模型

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deploy | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| modelId | String | | 模型Id | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "部署工作流模型成功" //返回的信息；

}

}

**e.功能实现说明**

1. 执行repositoryService接口的getMode方法，传入modelId参数，就可以得到这个模型。
2. 读这个模型的EditorSource文件放到ObjectNode中去，在把Object转化为BpmnModel，在从BpmnModel中得到byte[]数据。
3. 在把byte[]数据放到字符串bpmnString中去，从工作流模型得到模型名modelName，流程名processName设置为modelName加上‘.bpmn20.xml’的组合。
4. 执行repositoryService接口的.createDeployment() . name(modelName). addString(processName ,bpmnString) .deploy()方法.就可以部署好这个流程pi。
5. 从模型中得到配置的模型的id号sysModelId.
6. 执行processEngine.getIdentityService().newUser(pi.getd)方法,传入流程定义的Id号，创建了一个User对象，User对象的firstName设置模型的id号，User对象的lastName设置为sysModelId变量的值。
7. 创建Resultd对象，设置msg为部署流程成功，flag为0，并返回这个Result对象。
   * 1. 执行（完成）一个流程任务接口

**a.接口url:**

pub/admin/system/workFlow/completeTask.do

**b.接口说明：**

执行（完成）一个流程任务

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| completeTask | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| taskId | String | | 任务id | |
| jsonInfo | String | | 上传的Json数据 | |
| completeName | String | | 完成人 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "完成任务成功" //返回的信息；

}

}

**f.功能实现说明**

1. 根据任务id查到流程定义id，在查询activiti数据库的User表来得到系统模型id。
2. 执行updatData函数（该函数会返回信息id），传入jsonInfo，任务id,系统模型id三个参数根据任务id来得到任务往下走的逻辑变量list。
3. 遍历这个list，根据变量名来从jsonInfo中取数据，再把变量名当成key值，取得数据当成value值。放到一个map中。如果取不到数据，根据任务id调前面的数据来取值。
4. 让completeName人来拾取这个任务。
5. 把信息id绑定到这个任务上。
6. 在这个任务上设置一个qTaskId的参数，值为此任务的Id
7. 执行完成任务的方法，传入任务id和map。
8. 创建Resultd对象，设置msg为执行任务成功，flag为0，并返回这个Result对象。
   * 1. 启动（也可以叫申请或产生）一个新的流程接口

**a.接口url:**

pub/admin/system/workFlow/ startProcess.do

**b.接口说明：**

申请流程

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| startProcess | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| processkey | String | | 流程的key值 | |
| aName | String | | 申请人 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "申请成功" //返回的信息；

}

}

**e.功能实现说明**

1. 把参数aName放到map中。
2. 执行runtimeService().startProcessInstanceByKey(processkey,map)方法。
3. 传入参数processkey和map。
4. 创建Resultd对象，设置msg为申请成功，flag为0，并返回这个Result对象。
   * 1. 查询某人已完成或未完成的任务

**a.接口url:**

pub/admin/system/workFlow/decisionMakerTask.do

**b.接口说明：**

查询一个人已经或未完成的任务（根据节点id来查任务，根据决策人姓名来查任务或根据决策人姓名和流程定义id来查任务）

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| decisionMakerTask | | infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| nodeId | String | | 节点id | |
| processDefinitionId | String | | 流程定义id | |
| decisionMaker | String | | 决策人姓名 | |
| complete | Integer | | 0表示未完成的任务，1表示已经完成的任务 | |
| pageSize | Integer | | 每页尺寸 | |
| nowPage | Integer | | 当前页 | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [

{

"id": "", //任务Id

"assignee": "", //执行人

"createTime": "", //创建时间

"name": "", //姓名

"processDefinitionId": "", //流程定义id

"processInstanceId": "" , //流程实例id

"description": "", //描述

“inforId”:”” //信息Id

},

{}

]

}

**f.功能实现说明**

1. 当complete参数为0的时候。
   1. 如果nodeId不为空且decisionMake不为空，执行TaskService接口的createTaskQuery. taskDefinitionKey(nodeId). taskCandidateOrAssigned (decisionMaker) . listPage(nowPage, pageSize)方法
   2. 如果nodeId不为空，执行TaskService接口的createTaskQuery. taskDefinitionKey(nodeId) .listPage(nowPage, pageSize)方法.（包括一个流程下节点下所有的任务，比如申请的话，所有申请人任务都在）
   3. 如果decisionMake不为空且processDefinitionId不为空，执行TaskService接口的createTaskQuery. processDefinitionId(processDefinitionId) . taskCandidateOrAssigned(decisionMaker).listPage(nowPage,pageSize)方法.（一个流程下，一个人的所有任务，可能有其他路径）
   4. 如果decisionMake不为空，执行TaskService接口的createTaskQuery. taskCandidateOrAssigned (tdecisionMaker) .listPage(nowPage, pageSize)方法.（一个人所有的任务，可能多个流程）
   5. 往方法中传入tdecisionMaker,nowPage和pageSize这三个参数，把得到的数据放到infor对象中的infos字段。把得到数据的长度放到infor对象中的total字段中。
   6. 查找每一个任务的上一个任务绑定的信息id变量，放到infor对象的list变量的一个对象中。
   7. 执行相应的查询数量方法，把得到的数据放到infor对象中的page对象的lines变量中，在给page对象设置page,pages,pageSize,pageStart字段。
   8. 返回这个infor对象。
2. 当complete参数为1的时候。
   1. 如果nodeId不为空且decisionMake不为空,执 行historyService(). taskDefinitionKey(nodeId) . taskAssignee(decisionMaker). Finished ( ).listPage(nowPage\*pageSize-pageSize, nowPage\*pageSize);
   2. 如果nodeId不为空,执 行historyService(). taskDefinitionKey(nodeId) .finished().listPage(nowPage\*pageSize-pageSize, nowPage\*pageSize);
   3. 如果decisionMake不为空且processDefinitionId不为空执行historyService()..createHistoricTaskInstanceQuery().processDefinitionId(processDefinitionId).taskAssignee(decisionMaker).finished().listPage(nowPage\*pageSize-pageSize, nowPage\*pageSize);
   4. 如果decisionMake不为空,执 行historyService()..createHistoricTaskInstanceQuery().taskAssignee(decisionMaker).finished().listPage(nowPage\*pageSize-pageSize, nowPage\*pageSize);
   5. 往方法中传入tdecisionMaker,（nowPage\*pageSize-pageSize）和pageSize这三个参数，把得到的数据放到infor对象中的infos字段。把得到数据的长度放到infor对象中的total字段中。
   6. 执行historyService接口的相应的count()方法，把得到的数据放到infor对象中的page对象的lines变量中，在给page对象设置page,pages,pageSize,pageStart字段。
   7. 返回这个infor对象。
      1. 查询流程实例信息

**a.接口url:**

pub/admin/system/workFlow/sezrch HistoricProcessInstance.do

**b.接口说明：**

查询单个流程的基本信息

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| sezrchHistoricProcess Instance | | Map<String,Object> | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| processInstanceId | String | | 流程实例Id | |

**d.返回值示例与说明**

{

"processDefinitionId": "leaveProcess:1:4", //流程定义Id

"businessKey": null, //流程的key值

"name": null, //流程的名称

"description": null, //描述

"startTime": "Wed Sep 14 11:02:51 CST 2016", //流程的开始时间

"endTime": "Wed Sep 14 11:49:32 CST 2016", //流程的结束时间

"id": "2501" //流程的id

}

**f.功能实现说明**

1. 执行historyService接口的createHistoricProcessInstanceQuery () . processInstanceId(businessKey). singleResult();
2. 得到HistoricProcessInstance的数据。
3. 定义一个map，把HistoricProcessInstance的一些参数放到这个map中
4. 返回map。
   * 1. 查询流程实例的过程（任务列表）

**a.接口url:**

pub/admin/system/workFlow/searchTaskHis.do

**b.接口说明：**

查询一个流程的执行了哪些任务

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| searchTaskHis | | List<Map<String, Object>> | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| processInstanceId | String | | 流程实例Id | |

**d.返回值示例与说明**

[

{

"id": "", //任务Id

"assignee": "", //执行人

"createTime": "", //创建时间

"name": "", //姓名

"processDefinitionId": "", //流程定义id

"processInstanceId": "" , //流程实例id

"description": "", //描述

"endTime": "", //结束时间

}

]

**f.功能实现说明**

1. 执行historyService接口的createHistoricTaskInstanceQuery() . processInstanceId(processInstanceId). list();
2. 得到List<HistoricTaskInstance>的数据。
3. 再把List<HistoricTaskInstance>返回到页面
   * 1. 查询流程定义列表

**a.接口url:**

pub/admin/system/workFlow/sezrch ProcessDefinitions.do

**b.接口说明：**

查询已经部署的流程有哪些

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| sezrch ProcessDefinitions | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pageSize | Integer | | 每页尺寸 | |
| nowPage | Integer | | 当前页 | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [

{

"DeploymentId": "1", //部署Id

"id": "leaveProcess:1:4", //流程定义Id

"Key": "leaveProcess", //流程定义的Key

"Name": "leaveMyProcess" //流程名

} ,

{}

]

}

**f.功能实现说明**

1. 执行repositoryService接口的createProcessDefinitionQuery () .listPage(nowPage, pageSize);
2. 往方法中传入nowPage和pageSize这两个参数，把得到的数据放到infor对象中的infos字段。把得到数据的长度放到infor对象中的total字段中。
3. 执行repositoryService接口的createHistoricProcessInstanceQuery().count()方法，把得到的数据放到infor对象中的page对象的lines变量中，在给page对象设置page,pages,pageSize,pageStart字段。
4. 返回这个infor对象。
   * 1. 查询流程实例列表

**a.接口url:**

pub/admin/system/workFlow/sezrch HistoricProcessInstances.do

**b.接口说明：**

求一个流程产生了多少条流程实例

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| sezrchHistoricProcess Instances | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| processDefinitionId | String | | 流程定义id | |
| pageSize | Integer | | 每页尺寸 | |
| nowPage | Integer | | 当前页 | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [

{

"processDefinitionId": "leaveProcess:1:4", //流程定义Id

"name": null, //流程的名称

"description": null, //描述

"startTime": "Wed Sep 14 11:02:51 CST 2016", //流程的开始时间

"endTime": "Wed Sep 14 11:49:32 CST 2016", //流程的结束时间

"id": "2501" //流程的id

},

{}

]

}

**f.功能实现说明**

1. 执行HistoryService(). createHistoricProcessInstanceQuery(). processDefinitionId(processDefinitionId) .listPage(nowPage, pageSize)。
2. 往方法中传入processDefinitionId,nowPage和pageSize这三个参数，把得到的数据放到infor对象中的infos字段。把得到数据的长度放到infor对象中的total字段中。
3. 执行HistoryService接口的createHistoricProcessInstanceQuery(). processDefinitionId(processDefinitionId. count()方法，把得到的数据放到infor对象中的page对象的lines变量中，在给page对象设置page,pages,pageSize,pageStart字段。
4. 返回这个infor对象。
   * 1. 查找节点候选人名单接口

**a.接口url:**

pub/admin/system/workFlow/findCandidater.do

**b.接口说明：**

查询此节点候选人名单

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findCandidater | | List<Map<String,String>> | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| taskId | String | | 任务id | |

**d.返回值示例与说明**

[

{

"personName": "" //候选人姓名

},

{

"personName": "" //候选人姓名

}

]

**e.功能实现说明**

1. 执行taskService接口的getIdentityLinksForTask(taskId)方法。
2. 传入参数taskId，会得到IdentityLink对象的list，遍历list，把IdentityLink对象的UserId字段值赋值给personName变量，把personName变量放到一个map中，这个map中的数据放到list中
3. 返回list。
   * 1. 拾取任务接口

**a.接口url:**

pub/admin/system/workFlow/claimTask.do

**b.接口说明：**

拾取任务

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| claimTask | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| taskId | String | | 任务id | |
| userId | String | | 拾取人名称 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "拾取成功" //返回的信息；

}

}

**e.功能实现说明**

1. TaskService接口执行claim方法，传入taskId和userId两个变量。
2. 创建Resultd对象，设置msg为拾取成功，flag为0，并返回这个Result对象。
   1. 附件管理
      1. 上传单个文件

**a.接口url:**

pub/admin/system/enclosure/uploadFileOne.do

**b.接口说明**

上传单个附件，返回文件的信息

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| uploadFileOne | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| file | MultipartFile | | 上传的文件变量 | |
| filetype | String | | 文件类型，文件类型用数据词典做的 | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

" fileid ": "547a7578defe483b9d7629a1881cdb5b", //文件id;

" filename ": "1.jpg", //文件名;

" adduser ": "张三", //增加者;

" filesize ": "14000", //文件的大小;

" filetime ": "2016-08-13 16:16:08", //文件的增加时间;

“filetype”:””, //文件类型

“path”:””, //文件的路径

“classid”:””, //栏目id

“no”:””, //文件别名

“onclick”:””, //下载次数

“id”:””, //信息id

“cjid”：“”， //文件临时识别编码

“md5”：“”， //文件MD5值

“fid”：“”， //文件的父id

“ftype”：“”， //文件类型

}

]

}

**e.功能实现说明**

1. 把上传的判断是否存在，
   1. 存在就只存文件的信息到数据库
   2. 不存在，存文件信息到数据库中，在把文件存到服务器上
      1. 传附件和属性接口

**a.接口url:**

pub/admin/system/enclosure/uploadFile.do

**b.接口说明**

1. 上传单个附件或多个附件
2. 对上传图片可以生成缩略图

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| uploadPic | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| getsmall | String | | 生成缩略图标志（为1生成缩略图，其他不生成） | |
| no | String[] | | 给多个文件设置别名，不写的文件别名设置成原始文件名 | |
| width | Integer | | 生成缩略图图片的宽度（默认值200） | |
| height | Integer | | 生成缩略图图片的高度（默认值200） | |
| file | MultipartFile[] | | 上传的文件变量 | |
| classid | string | | 所属栏目id、可为空 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "上传附件成功" //返回的信息；

}

}

**e.功能实现说明**

1. 对file进行遍历，假如其中有的文件是空的，结束本次循环，进行下一次循环。遍历结束，跳到第8步。
2. 取得服务器上工程的根路径。
3. 首先计算文件MD5值，搜索数据库这个MD5值是否存在，如果不存在，继续执行下一步。若果存在，跳过第4步，从第5步执行。
4. 把附件的文件名改为MD5值，在把这个附件存储到根路径下pub/admin/upload中。（备注：存储附件的时候，先建一个日期文件夹，再来存储附件，日期文件夹用年月日来命名：比如20160817，如果日期文件夹存在，就不需要再建，附件就直接放到这个日期文件夹下）。
5. 在new一个zjxxEnewsfile对象，给对象字段设置值（**注意新增了classid字段**），设置好了把对象对象存储到数据库中。（如果这个文件的MD5值已经存在，zjxxEnewsfile对象的path字段从MD5相同的一条数据中得到）。
6. 假如getsmall值是1，根据width 和 height把图片文件生成缩略图。在执行3,4,5步，用来保存缩略图数据和附件，在执行第5步的时候，必须要设置fid和ftype这两个字段，fid表示这张缩略图元文件的id，ftype设置为0字符串。（fid和ftype用来设置缩略图与原图的逻辑）结束后执行第7步。
7. 跳到第1步。直到遍历结束。
8. 创建Resultd对象。
9. 如果上传成功，设置msg为上传附件成功，flag为0，否则设置msg为上传附件失败，flag为1.
10. 返回这个Result对象。
    * 1. 查找附件列表接口

**a.接口url:**

pub/admin/system/enclosure/searchFile.do

**b.接口说明：（**nowPage和pageSize两个参数必须传**）**

1. 搜索附件列表接口
2. 条件搜索附件
   * 1. 根据文件夹名来搜索附件，在加上folderName参数
     2. 根据文件名，编号，上传者条件来搜索附件，在加上filename或no或adduser参数。

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| searchFile | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| nowPage | Integer | | 当前页 | |
| pageSize | Integer | | 每页显示的尺寸 | |
| folderName | String | | 文件夹名 | |
| filename | String | | 文件名 | |
| no | String | | 编号 | |
| adduser | String | | 上传者 | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

" fileid ": "547a7578defe483b9d7629a1881cdb5b", //文件id;

" filename ": "1.jpg", //文件名;

" adduser ": "张三", //增加者;

" filesize ": "14000", //文件的大小;

" filetime ": "2016-08-13 16:16:08", //文件的增加时间;

“path”:””, //文件的路径

“classid”:””, //栏目id

“no”:””, //文件别名

“onclick”:””, //下载次数

“id”:””, //信息id

“cjid”：“”， //文件临时识别编码

“md5”：“”， //文件MD5值

“fid”：“”， //文件的父id

“ftype”：“”， //文件类型

},

{}

]

}

**e.功能实现说明**

1. 把（nowPage\*pageSize-nowPage），pageSize，folderName, no, adduser,filename这六个参数放入map中。
2. 执行dao层的查询数据方法，把得到的数据放到infor对象的list中。把得到数据的长度放到infor对象的total字段中。
3. 执行dao层的查询数据长度方法，传入map，把得到的数据放到infor对象的page对象的lines中。在给page对象设置page,pages,pageSize,pageStart字段。
4. 返回这个infor对象
   * 1. 附件下载

**a.接口url:**

pub/admin/system/enclosure/Downloadfile.do

**b.接口说明：**

下载附件到本地

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| Downloadfile | | ResponseEntity<byte[]> | | 二进制流 |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| fileid | String | | 文件的id | |

**d.返回值示例与说明**

返回文件

**e.功能实现说明**

1. 根据文件的id，执行dao层方法，传入id。得到zjxxEnewsfile对象
2. 从zjxxEnewsfile中的到onclick字段值，加上1，在设置到zjxxEnewsfile对象的onclick字段中。
3. 执行dao层的修改方法，传入zjxxEnewsfile参数。
4. 从zjxxEnewsfile对象中的得到文件的路径path。
5. 取得服务器上的工程路径和pub/admin/uploa和path组成一个文件的路径，从而得到这个文件
6. ResponseEntity<byte[]>方法把文件以二进制流的方式写出去。
   * 1. 附件查看

**a.接口url:**

pub/admin/system/enclosure/lookfilejson.do

**b.接口说明：**

返回文件的具体信息

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| lookfilejson | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| fileid | String | | 文件的id | |

**d.返回值示例与说明**

{

“infos”:

{

" fileid ": "547a7578defe483b9d7629a1881cdb5b", //文件id;

" filename ": "1.jpg", //文件名;

" adduser ": "张三", //增加者;

" filesize ": "14000", //文件的大小;

" filetime ": "2016-08-13 16:16:08", //文件的增加时间;

“path”:””, //文件的路径

“classid”:””, //栏目id

“no”:””, //文件别名

“onclick”:””, //下载次数

“id”:””, //信息id

“cjid”：“”， //文件临时识别编码

“md5”：“”， //文件MD5值

“fid”：“”， //文件的父id

“ftype”：“”， //文件类型

}

}

**e.功能实现说明**

1. 根据文件的id，执行dao层方法，传入id。得到zjxxEnewsfile对象
2. 返回zjxxEnewsfile对象
   * 1. 查询根级目录接口

**a.接口url:**

pub/admin/system/enclosure/searchCatalog.do

**b.接口说明：**

搜索目录

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| searchCatalog | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pageSize | Integer | | 每页尺寸 | |
| nowPage | Integer | | 当前页 | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

" folderName": "201415", //文件夹名;

}

]

}

**e.功能实现说明**

1. 先取得服务器上工程的根路径，把根路径和pub/admin/upload组成一个新路径path2.
2. 取得path2路径下的文件数组m，遍历这个文件数组，假如此文件是一个文件夹，把这个文件放到文件数组a中
3. 把a数组的长度放到infor对象的page对象中的lines字段中。
4. 取a数组中（nowPage\*pageSize-nowPage）到（nowPage\*pageSize）的数据放到list<map>中，每一个map的key值为folderName。
5. 在把list<map>设置到infor对象的list中，把list<map>的长度设置到infor对象的total字段中。
6. 给page设置其他字段。
7. 返回这个infor对象.
   * 1. 裁剪图片接口

**a.接口url:**

pub/admin/system/enclosure/cutImage.do

**b.接口说明：**

裁剪图片

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| cutImage | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| fileid | String | | 文件id | |
| width | Integer | | 剪切文件的宽度 | |
| height | Integer | | 剪切文件的高度 | |
| x | Integer | | 剪切图片起始点x坐标 | |
| y | Integer | | 剪切图片起始点y坐标 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "剪切图片成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据文件id的来得到文件的存储路径，从而得到这个文件。
2. 根据width，height,x和y对这张图片进行剪切，得到剪切后的文件file2.
3. 计算文件file2的MD5值，搜索数据库这个MD5值是否存在，如果不存在，继续执行下一步。若果存在，跳过第4步，从第5步执行。
4. 把file2附件存储到根路径下pub/admin/upload文件夹中。（备注：存储附件的时候，先建一个日期文件夹，再来存储附件，日期文件夹用年月日来命名：比如20160817，如果日期文件夹存在，就不需要再建，附件就直接放到这个日期文件夹下）
5. 在new一个zjxxEnewsfile对象，给对象字段设置值，必须设置fid和ftype这两个字段，fid的值为fileid，ftyppe的值为1。（fid，ftype用来设置剪切图与原图的逻辑）设置好了把对象对象存储到数据库中。（如果这个文件的MD5值已经存在，zjxxEnewsfile对象的path字段从MD5相同的一条数据中得到）
6. 创建Resultd对象。
7. 如果剪切成功，设置msg为剪切成功，flag为0，否则设置msg为剪切失败，flag为1.
8. 返回这个Result对象。
9. 栏目
   1. 栏目管理
      1. 查看栏目列表接口：

**a.接口url:**

pub/admin/ column/manageColumn/searchColumn.do

**b.接口说明：（**pageSize和nowPage两个参数必须要传**）**

1. 查看每一级栏目数据
   * 1. 查根级栏目，在传bclassid参数，值为0字符串
     2. 查子级栏目，在传bclassid参数，值为当前栏目id
2. 条件搜索栏目数据
   * 1. 在传classid或classname或bname或bclassid这四个参数

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| searchColumn | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| classid | String | | 栏目Id | |
| pageSize | Integer | | 每页尺寸 | |
| nowPage | Integer | | 当前页 | |
| classname | String | | 栏目名称 | |
| bname | String | | 栏目别名 | |
| bclassid | String | | 父栏目id | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [

{

"classid": "sk", //栏目id

"classname": "sf", //栏目名

"onclick": "sfasf", //访问量

"myorder": "sfdf", //顺序

"onclick": "", //点击数

"islast": "", //是否为终极栏目

"bname": "", //栏目别名

"columntype": "", //栏目类型

"isgd":"", //是否归档

"isinfo": "", //是否自动转换为信息

"showclass": "", //是否显示到导航

"openadd": "", //是否开放投稿

"columnclass":"", //栏目页模式

"showdt":"", //内容页模式

"isflash":"", //内容条数增删时，列表页是否自动刷新

"maxnum":"", //显示总记录数

"lencord":"", //每页显示信息数

"linknum":"", //相关链接数

"classtype": "", //栏目文件扩展名

"filetype": "", //内容页文件扩展名

"author":"", //作者

"top": "", //是否置顶

"downnum": "", //每行下载地址数

"onlinenum": "", //每行在线观看地址数

"addinforfen":"", //前台投稿加点数

"listdt": "", //列表模式

"doctime": "", //信息归档时间

"stpnum": "", //生成静态页数

"addreinfor": "", //发布信息生成内容页

"modid": "", //为所属系统模型ID

"isshow": "", //是否显示

“pid”：“”， //父栏目id

"openpl":"", //是否开启评论

"plmodids":"", //评论系统模型id集合

"orgid":"", //所属机构id

"trainfo":"", //转换为信息

"columnid": "" //自动转换信息栏目

},

{

}

]

}

**e.功能实现说明**

1. 把(nowPage\*pageSize-pageSize),pagesize.classid，classname, bname, bclassid这6个参数放到map中。
2. 执行dao层接口的查找方法，传入map变量，把得到数据设置到infor对象的infos字段中，数据的长度设置到infor对象的total字段中
3. 执行dao层接口的查找数据总长度方法，传入map变量，把得到的数据放到infor对象中的page对象的lines变量中，在给page对象设置page,pages,pageSize,pageStart字段。
4. 返回infor对象。
   * 1. 删除栏目接口

**a.接口url:**

pub/admin/column/manageColumn/deleteColumn.do

**b.接口说明：**

删除栏目信息

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteColumn | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| classid | String | | 栏目的id | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "删除栏目成功" //返回的信息；

}

**e.功能实现说明**

1. 根据栏目的classid查找出栏目主表一条记录zjxxEnewscolumn,从记录中的得到系统模型modid。
2. 根据modid来查找模型表的记录，得到ZjxxModel数据。
3. 得到ZjxxModel数据中的数据表名，执行dao层接口的删除信息表数据的方法，传入数据表名和classid来删除数据。
4. 用递归的方法来删除当前栏目后代的记录。（栏目有四张表，一张主表和三张副表，要执行4个删除方法）
5. 创建Resultd对象，设置msg为删除栏目成功，flag为0，并返回这个Result对象。
   * 1. 增加/修改栏目接口

**a.接口url:**

pub/admin/ column/manageColumn/addOrUpdateColumn.do

**b.接口说明：**

增加或修改栏目的数据

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| addOrUpdateColumn | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| z1 | ZjxxEnewscolumn对象 | | 栏目主表 | |
| p1 | ZjxxEnewsColumnPart1对象 | | 栏目副表 | |
| p2 | ZjxxEnewsColumnPart2对象 | | 栏目副表 权限 | |
| p3 | ZjxxEnewsColumnPart3对象 | | 栏目副表 大字段 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "修改成功" //返回的信息；

}

**e.功能实现说明**

1. 根据对象参数z1，判断对象中classid字段值是否为空：
   * 1. 创建Result对象。
     2. 假如classid为空：
        1. 设置Uuid给对象中z1,p1,p2,p3的classid。
        2. 执行dao层接口的增加方法，增加数据库的数据。（有4个对象，要执行4个增加方法
        3. 如果增加成功，设置msg为增加栏目成功，flag为0，否则设置msg为增加栏目失败，flag为1。
     3. 假如classid为不为空：
        1. 执行dao层接口的修改方法，修改数据库的数据。（有4个对象，要执行4个修改方法）
        2. 如果修改成功，设置msg为添加栏目成功，flag为0，否则设置msg为修改栏目失败，flag为1.
     4. 返回Result对象。
     5. 查找关联字段

**a.接口url:**

pub/admin/ column/manageColumn/ findRelatedFields.do

**b.接口说明：**

查找关联字段

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findRelatedFields | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| modid | String | | 模型id | |

**d.返回值示例与说明**

{

“infos”：

[

{

"field": "zjxx\_a2" //关联字段

},

{

"field": "aa" //关联字段

}]

}

**e.功能实现说明**

1. 根据modid查询这个模型属于哪张表。
2. 搜索这张表每个字段的关联字段和这张表的分表中的关联字段。把这些关联字段放到list<map>中，一个关联字段一个map，（其中map的key值为field）。
3. 用递归的方法查找这张表的父表和父表的分表，直到这张表是父表为空。把这些父表和父表的分表中的关联字段写到list<map>中，一个关联字段一个map，（其中map的key值为field）。
4. 返回info。
   * 1. 修改栏目顺序接口

**a.接口url:**

pub/admin/ column/manageColumn/updateColumnOrder.do

**b.接口说明：**

修改栏目顺序的接口

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| updateColumnOrder | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| myorder | String[] | | 顺序值的数组 | |
| classid | String[] | | 栏目Id的数组（顺序值和栏目Id是一一对应） | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "修改成功" //返回的信息；

}

**e.功能实现说明**

1. 创建Resultd对象。
2. 遍历两个数组，新建一个map变量，把两个数组对应的值放到map中，key分别为myorder，calssid。执行栏目主表对象的对应dao层的接口的修改方法，修改数据库的数据。
3. 如果修改成功，设置msg为修改栏目顺序成功，flag为0，如果遍历中抛异常设置msg为修改栏目顺序失败，flag为1.
4. 返回这个Result对象。
   * 1. 查询绑定模板的栏目

**a.接口url:**

pub/admin/ column/manageColumn/findByTempid.do

**b.接口说明：**

传入模板的id，查询有没有栏目绑定该模板

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| FindByTempid | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| listtempid | String | | 列表模板id | |
| covertempid | String | | 封面模板id | |
| contenttempid | String | | 内容模板Id | |

**d.返回值示例与说明**

{

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

"classid": "sk", //栏目id

"classname": "sf", //栏目名

"onclick": "sfasf", //访问量

"myorder": "sfdf", //顺序

"onclick": "", //点击数

"islast": "", //是否为终极栏目

"bname": "", //栏目别名

"columntype": "", //栏目类型

"isgd":"", //是否归档

"isinfo": "", //是否自动转换为信息

"showclass": "", //是否显示到导航

"openadd": "", //是否开放投稿

"columnclass":"", //栏目页模式

"showdt":"", //内容页模式

"isflash":"", //内容条数增删时，列表页是否自动刷新

"maxnum":"", //显示总记录数

"lencord":"", //每页显示信息数

"linknum":"", //相关链接数

"classtype": "", //栏目文件扩展名

"filetype": "", //内容页文件扩展名

"author":"", //作者

"top": "", //是否置顶

"downnum": "", //每行下载地址数

"onlinenum": "", //每行在线观看地址数

"addinforfen":"", //前台投稿加点数

"listdt": "", //列表模式

"doctime": "", //信息归档时间

"stpnum": "", //生成静态页数

"addreinfor": "", //发布信息生成内容页

"modid": "", //为所属系统模型ID

"isshow": "", //是否显示

“pid”：“”， //父栏目id

"openpl":"", //是否开启评论

"plmodids":"", //评论系统模型id集合

"orgid":"", //所属机构id

"trainfo":"", //转换为信息

"columnid": "" //自动转换信息栏目

},

“result” : { //结果集;

"flag": "1", //标志，0为成功，1为操作失败

"msg": "有栏目绑定" //返回的信息；

}

]

}

**e.功能实现说明**

1. 创建Resultd对象。
2. 根据传入的参数查询绑定了模板的栏目id和传入的模板id相一致的栏目
3. 如果listtempid不为空时拼接查找条件，查询栏目表中列表模板id为传入的listtempid的模板
4. 如果covertempid不为空时拼接查找条件，查询栏目表中列表模板id为传入的covertempid的模板
5. 如果contenttempid不为空时拼接查找条件，查询栏目表中列表模板id为传入的contenttempid的模板

3）判断有没有找到栏目，根据查找情况设置infor信息病房result

* + 1. 栏目转换信息接口

**a.接口url:**

pub/admin/ column/manageColumn/columnTransferToinfo.do

**b.接口说明:**

将信息数据转换为栏目

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| infoTransferTocolumn | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| classid | String | | 栏目的id | | |
| classname | String | | 栏目的名字 | | |
| tbname | String | | 转换目标的表名 | | |
| fid | String | | 选择信息所属的栏目id | | |

**d.返回值示例与说明**

{

"flag": "1", //返回值标志

"msg": "转移成功", //返回值信息

}

**e.功能实现说明**

1. 接收参数classid、classnamee、fid，判断有任意一个空值则返回“来源错误”。
2. 根据classid获取来源数据，若找不到数据则同样返回“来源错误”。
3. 获取来源字段fid，获取栏目对应的系统模型。
4. 根据系统模型获得对应的信息表名。
5. 将获取的信息表名作为tbname的值，classid、classname作为信息表对应的字段拼接sql语句。
6. 执行sql，往信息表里插入一条记录。
7. 返回result。
8. 模版
   1. 列表模版
      1. 查找列表模版列表接口

**a.接口url:**

/Pub/admin/template/listTemp/findListTempList.do

**b.接口说明:**

该接口用于根据参数查询对应列表模板信息

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findListTempList | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pageSize | int | | 每页显示信息数 | |
| nowpage | Int | | 当前页 | |
| tempname | String | | 输入的模版名称 | |
| modid | String | | 系统模型ID | |
| orgname | String | | 机构名称 | |
| orgid | String | | 机构ID | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

"tempid": "8128", //模版ID;

"orgid": "6c33", //机构ID;

"tempname": "新闻列表模版", //模版名称 ;

"temptext": "html", //模版内容;

"subnews": "2", //简介截取字数;

"isdefault": "0", //是否默认模版;

"listvar": "rd", //列表内容模板;

"rownum": "20", //每次显示;

"modid": "232", //系统模型ID;

"showdate": "yy-dd-hh", //时间显示格式;

"subtitle": "10", //标题截取字段数;

"docode": "0" //是否使用远程代码;

},

{}

]

}

**e.功能实现说明**

1. 根据参数tempname,modid，orgname，orgid查找列表模版：

① orgname不为空时拼接查找条件，查找机构表中所有title字段值like参数 orgname的数据的id，以id为条件查找列表模版表中orgid值为id值的数据;

② orgid不为空时拼接查找条件，查找列表模版表中orgid值为参数orgid值的 数据;

③ tempname不为空时拼接查找条件,查找列表模版表中tempname字段值like当前参数tempname值的数据;

④ modid不为空时拼接查找条件，查找列表模版中modid字段值为参数modid值的数据;

⑤ 根据参数nowpage，pageSize计算出查找条件，追加查找条件limit;

⑥ 根据拼接的sql语句执行查询操作，返回查询列表，并根据返回的数据列表，分页实体类Page,返回信息类Info整合数据。

注：page对象字段详细情况参照6.6.1分页类对象，info对象的字段详细信息参照6.6.2返回信息类对象。

* + 1. 查找列表模版信息接口

**a.接口url:**

/Pub/admin/template/listTemp/findListTempByTempid.do

**b.接口说明:**

该接口用于根据列表模版ID查找列表模版

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findListTempByTempid | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| tempid | String | | 模版ID | |

**d.返回值示例与说明**

{

“infos”:

{

"tempid": "8128", //模版ID;

"orgid": "6c33", //机构ID;

"tempname": "新闻列表模版", //模版名称 ;

"temptext": "html", //模版内容;

"subnews": "2", //简介截取字数;

"isdefault": "0", //是否默认模版;

"listvar": "rd", //列表内容模板;

"rownum": "20", //每次显示;

"modid": "232", //系统模型ID;

"showdate": "yy-dd-hh", //时间显示格式;

"subtitle": "10", //标题截取字段数;

"docode": "0" //是否使用远程代码;

}

}

**e.功能实现说明**

(1)根据参数tempid查找列表模版表中tempid值为参数tempid值的数据。

* + 1. 删除列表模版接口

**a.接口url:**

/Pub/admin/template/listTemp/deleteListTemp.do

**b.接口说明:**

该接口用于根据模版ID删除列表模板信息

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteListTemp | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| tempid | String | | 列表模版ID | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "删除列表模版成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数tempid删除列表模版，判断有没有栏目绑定了该模板：

若有给出提示有栏目绑定该模板是否删除，确认删除则直接根据tempid删除该模版

~~① 若有栏目绑定了该模板，给出提示更改栏目绑定的模版，即更新绑定了该模板的栏目数据，修改其为更改后的模版，刷新缓存表中该模版生成的页面，根据参数tempid删除列表模版表中该模版数据;~~

~~② 若没有栏目绑定该模板，直接根据参数tempid删除列表模版表中该数据。~~

* + 1. 增加/修改列表模版接口

**a.接口url:**

/Pub/admin/template/listTemp/addListTemp.do

**b.接口说明:**

该接口用于增加或修改列表模板信息

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| addListTemp | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| listTemp | ZjxxListTemp (对象) | | 列表模版类对象,对象详细参数参照文档6.3.1 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "修改列表模版成功" //返回的信息；

}

}

或者

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "增加列表模版成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据对象参数listTemp 判断对象中tempid字段值是否为空：
2. tempid值为空时，生成UUID赋值给对象中tempid字段，然后将对象插入列表模版表中;
3. tempid字段不为空时：

1） 根据对象中tempid的值查找列表模版表中tempid字段值为对象中tempid 值的数据，对该数据进行更新，更新的字段及字段值为对象参数中对应的字段 和字段值。

2）~~刷新缓存表中使用了该模板生成的页面。~~

* 1. 内容模版
     1. 分查找内容模版列表接口

**a.接口url:**

/Pub/admin/template/contentTemp/findContentTempList.do

**b.接口说明:**

该接口用于根据参数查询内容模板信息

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findContentTempList | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pageSize | int | | 每页显示信息数 | |
| nowpage | Int | | 当前页 | |
| modid | String | | 系统模型ID | |
| orgname | String | | 机构名称 | |
| tempname | String | | 模版名称 | |
| orgid | String | | 机构ID | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

"tempid": "887sds", //模板ID;

"tempname": "内容模版", //模板名;

"modid": "df331", //所属系统模型ID;

"orgid": "sdfs d3453534", //机构ID;

"showdate": "Y-m-dH:i:s", //时间显示格式;

"temptext": "12", //模板内容;

"isdefault": 0 //是否默认模板;

},

{ }

]

}

**e.功能实现说明**

(1)根据参数tempname,orgname,modid查找内容模版：

1. orgname不为空时增加查找条件,查找机构表中所有title字段值like参数 orgname的数据的id，以id为条件查找内容模版表中orgid值为id值的数据;
2. orgid不为空时拼接查找条件，查找内容模版表中orgid值为参数orgid值 的数据。
3. tempname不为空时增加查找条件,查找内容模版表中tempname字段值like当 前参数tempname值的数据;
4. modid不为空时增加查找条件，查找内容模版中modid字段值为参数modid值 的数据;
5. 根据参数nowpage，pageSize计算出查找条件，追加查找条件limit;

⑥ 最后根据拼接的sql语句执行查询操作，返回查询列表，并根据返回的数据列表，分页实体类Page,返回信息类Info整合数据。

注：page对象字段详细情况参照6.6.1分页类对象，info对象的字段详细信息参照6.6.2返回信息类对象。

* + 1. 查找内容模版信息接口

**a.接口url:**

/Pub/admin/template/listTemp/findContentTempByTempid.do

**b.接口说明:**

该接口用于根据内容模版ID查找内容模版

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findContentTempByTempid | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| tempid | String | | 模版ID | |

**d.返回值示例与说明**

{

“infos”：

{

"tempid": "887sds", //模板ID;

"tempname": "内容模版", //模板名;

"modid": "df331", //所属系统模型ID;

"orgid": "sdfs d3453534", //机构ID;

"showdate": "Y-m-dH:i:s", //时间显示格式;

"temptext": "12", //模板内容;

"isdefault": 0 //是否默认模板;

}

}

**e.功能实现说明**

(1)根据参数tempid查找内容模版表中tempid值为参数tempid值的数据。

* + 1. 删除内容模版接口

**a.接口url:**

/Pub/admin/template/contentTemp/deleteContentTemp.do

**b.接口说明:**

该接口用于根据模版ID删除内容模板信息

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteContentTemp | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| tempid | String | | 内容模版ID | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "删除内容模版成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数tempid删除内容模版，判断有没有栏目绑定了该模板：

若有给出提示有栏目绑定该模板是否删除，确认删除则直接根据tempid删除该模版

~~① 若有栏目绑定了该模板，给出提示更改栏目绑定的模版，即更新绑定了该模板的栏目数据，修改其为更改后的模版，刷新缓存表中该模版生成的页面，根据参数tempid删除内容模版表中该模版数据;~~

~~② 若没有栏目绑定该模板，直接根据参数tempid删除内容模版表中该模版数~~据。

* + 1. 增加/修改内容模版接口

**a.接口url:**

/Pub/admin/template/contentTemp/addContentTemp.do

**b.接口说明:**

该接口用于增加/修改内容模版用

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| addContentTemp | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| contentTemp | ZjxxContentTemp (对象) | | 内容模版类对象,对象详细参数参照文档6.3.2 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "增加内容模版成功" //返回的信息；

}

}

或

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "修改内容模版成功" //返回的信息；

}

}

**e.功能实现说明**

（1）根据对象参数contentTemp 判断对象中tempid字段值是否为空：

① tempid值为空时，生成UUID赋值给对象中tempid字段，然后将对象插入内容 模版表中;

② tempid字段不为空时：

1） 根据对象中tempid的值查找内容模版表中tempid字段值为对象中tempid 值的数据，对该数据进行更新，更新的字段及字段值为对象参数中对应的字段 和字段值。

2）~~刷新缓存表中使用了该模板生成的页面。~~

* 1. 封面模版
     1. 查找封面模版列表接口

**a.接口url:**

/Pub/admin/template/coverTemp/findCoverTempList.do

**b.接口说明:**

该接口用于根据参数查询对应封面模板信息

**c. 接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findCoverTempList | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pageSize | int | | 每页显示信息数 | |
| nowpage | Int | | 当前页 | |
| tempname | String | | 模版名称 | |
| orgname | String | | 机构名称 | |
| orgid | String | | 机构ID | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

"tempid": "wedff887sds", //模板ID;

"tempname": "新闻封面模版", //模板名称;

"temptext": "123", //模板内容;

"orgid": "sdsxcsd344" //机构ID;

},

{}

]

}

**e.功能实现说明**

(1)根据参数tempname,orgname查找封面模版：

1. orgname不为空时增加查找条件,查找机构表中所有title字段值like参数 orgname的数据的id，以id为条件查找封面模版表中orgid值为id值的数据;
2. orgid不为空时拼接查找条件，查找封面模版表中orgid值为参数orgid值 的数据。
3. tempname不为空时增加查找条件,查找封面模版表中tempname字段值like当 前参数tempname值的数据;
4. 最后根据参数nowpage，pageSize计算出查找条件，追加查找条件limit;

⑤ 根据拼接的sql语句执行查询操作，返回查询封面，并根据返回的数据封面，分页实体类Page,返回信息类Info整合数据。

注：page对象字段详细情况参照6.6.1分页类对象，info对象的字段详细信息参照6.6.2返回信息类对象。

* + 1. 查找封面模版信息接口

**a.接口url:**

/Pub/admin/template/listTemp/findCoverTempByTempid.do

**b.接口说明:**

该接口用于根据封面模版ID查找封面模版

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findCoverTempByTempid | | ZjxxCoverTemp | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| tempid | String | | 模版ID | |

**d.返回值示例与说明**

{

"tempid": "wedff887sds", //模板ID;

"tempname": "新闻封面模版", //模板名称;

"temptext": "123", //模板内容;

"orgid": "sdsxcsd344" //机构ID;

}

**e.功能实现说明**

(1)根据参数tempid查找封面模版表中tempid值为参数tempid值的数据。

* + 1. 删除封面模版接口

**a.接口url:**

/Pub/admin/template/coverTemp/deleteCoverTemp.do

**b.接口说明:**

该接口用于删除封面模版用

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteContentTemp | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| tempid | String | | 封面模版ID | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "删除封面模版成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数tempid删除封面模版，判断有没有栏目绑定了该模板：

若有给出提示有栏目绑定该模板是否删除，确认删除则直接根据tempid删除该模版

~~① 若有栏目绑定了该模板，给出提示更改栏目绑定的模版，即更新绑定了该模板的栏目数据，修改其为更改后的模版，刷新缓存表中该模版生成的页面，根据tempid删除封面模版表中该数据；~~

~~② 若没有栏目绑定该模板，根据传入的参数tempid,删除封面模版表中tempid字段值为传入参数tempid的这条数据;~~

* + 1. 增加/修改封面模版接口

**a.接口url:**

/Pub/admin/template/coverTemp/addCoverTemp.do

**b.接口说明:**

该接口用于增加/修改封面模版用

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| addCoverTemp | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| coverTemp | ZjxxCoverTemp (对象) | | 封面模版类对象,对象详细参数参照文档6.3.3 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "增加封面模版成功" //返回的信息；

}

}

或

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "修改封面模版成功" //返回的信息；

}

}

**e.功能实现说明**

（1） 根据对象参数coverTemp 判断对象中tempid字段值是否为空：

① tempid值为空时，生成UUID赋值给对象中tempid字段，然后将对象插入封面 模版表中;

② tempid字段不为空时：

1） 根据对象中tempid的值查找封面模版表中tempid字段值为对象中tempid 值的数据，对该数据进行更新，更新的字段及字段值为对象参数中对应的字段 和字段值。

2）~~刷新缓存表中使用了该模板生成的页面。~~

1. 用户
   1. 机构管理
      1. 查找机构/部门列表接口

**a.接口url:**

/pub/admin/user/organization/findOrgList.do

**b.接口说明:**

该接口用于根据参数查询机构（部门）列表

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值 |
| findOrgList | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pageSize | int | | 每页显示信息数 | |
| nowPage | Int | | 当前页 | |
| keyword | String | | 输入的机构或部门名称 | |
| parentOrg | String | | 父机构ID | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

"id": "4b", //机构ID;

"title": "子杰信息", //机构名称;

"parentOrg": "", //父机构ID；

"createDate": "", //创建时间；

"orgAddress": "11", //机构地址;

"orgEmail": "11", //机构email;

"orgPhone": "", //机构电话;

"isdelete": 0, //是否删除;

"legalOrg": "", //机构法人;

"orgSize": 11, //机构规模;

"description": "", //机构描述;

"childOrg": "" //子级机构ID集合，多个逗号隔开

},

{}

]

}

**e.功能实现说明**

1. 根据参数orgid,keyword,parentOrg查找机构（部门）：

① 拼接查找条件，查找机构表中parent\_org字段值为参数parentOrg值的数据;

② 当keyword不为空时拼接查找条件，查找机构表中title字段的值like参数keyword值的数据;

③ 拼接查找条件，查找机构中isdelete字段值为0的数据;

④根据参数nowpage，pageSize计算出查找条件，追加查找条件limit;

⑤ 最后根据拼接的sql语句执行查询操作，返回查询列表，并根据返回的数据列表，分页实体类Page,返回信息类Info整合数据。

注：page对象字段详细情况参照6.6.1分页类对象，info对象的字段详细信息参照6.6.2返回信息类对象。

* + 1. 查找机构/部门及其所有子孙部门列表信息接口

**a.接口url:**

/pub/admin/user/organization/findOrgListInfo.do

**b.接口说明:**

该接口用于查询机构以及该机构的子孙机构

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值 |
| findOrgListInfo | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构ID | |

**d.返回值示例与说明**

{

“infos”：

[{

"id": "4b", //机构ID;

"title": "子杰信息", //机构名称;

"parentOrg": "", //父机构ID；

"createDate": "", //创建时间；

"orgAddress": "11",‘’ //机构地址;

"orgEmail": "11", //机构email;

"orgPhone": "", //机构电话;

"isdelete": 0, //是否删除;

"legalOrg": "", //机构法人;

"orgSize": 11, //机构规模;

"description": "", //机构描述;

"childOrg": "" //子级机构ID集合，多个逗号隔开

},

{}

]

}

**e.功能实现说明**

(1)根据参数orgid查找机构（部门）：

① 查找机构表中id字段值为orgid字段值的数据,然后再根据查找到的数据id递归查询机构表中parent\_org字段值为数据id的数据;

* + 1. 查找子机构/部门列表接口 （与4.1.1功能重复）

**a.接口url:**

/pub/admin/user/organization/findOrgListInfo1.do

**b.接口说明:**

该接口用于查询机构下子机构（部门）列表的信息

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值 |
| findOrgListInfo1 | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| parentOrg | String | | 父机构ID | |

**d.返回值示例与说明**

{

“infos”:

{

"id": "4b", //机构ID;

"title": "子杰信息", //机构名称;

"parentOrg": "", //父机构ID；

"createDate": "", //创建时间；

"orgAddress": "11", //机构地址;

"orgEmail": "11", //机构email;

"orgPhone": "", //机构电话;

"isdelete": 0, //是否删除;

"legalOrg": "", //机构法人;

"orgSize": 11, //机构规模;

"description": "", //机构描述;

"childOrg": "" //子级机构ID集合，多个逗号隔开

}

}

**e.功能实现说明**

(1)根据参数parentOrg查找机构（部门）：

查找机构表中parent\_org字段值为parentOrg字段值的数据;

在返回到页面

* + 1. 查找机构/部门信息接口 +

**a.接口url:**

/pub/admin/user/organization/findOrgById.do

**b.接口说明:**

该接口用于根据机构ID查找机构（部门）详细信息

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值 |
| findOrgById | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构ID | |

**d.返回值示例与说明**

{

“infos”:

{

"id": "4b", //机构ID;

"title": "子杰信息", //机构名称;

"parentOrg": "", //父机构ID；

"createDate": "", //创建时间；

"orgAddress": "11", //机构地址;

"orgEmail": "11", //机构email;

"orgPhone": "", //机构电话;

"isdelete": 0, //是否删除;

"legalOrg": "", //机构法人;

"orgSize": 11, //机构规模;

"description": "", //机构描述;

"childOrg": "" //子级机构ID集合，多个逗号隔开

}

}

**e.功能实现说明**

（1）根据参数orgid查找机构表中id字段值为参数orgid并且该数据的isdelete字段值为0的数据。

* + 1. 修改机构/部门顺序接口

**a.接口url:**

/pub/admin/user/organization/updateOrgOrder.do

**b.接口说明:**

该接口用于修改部门

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| updateOrg | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| idStr | String[] | | 遍历页面数据的id的字符串数组 | |
| orderStr | String[] | | 遍历页面数据顺序的得到的字符串数组 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "修改顺序成功" //返回的信息；

}

}

**e.功能实现说明**

（1）根据参数idStr ，orderStr ，分别遍历数组，修改机构表中对应的数据的顺序字段值。

* + 1. 删除机构/部门接口

**a.接口url:**

/pub/admin/user/organization/deleteOrg.do

**b.接口说明:**

该接口用于删除或批量删除机构或部门用

**c.接口定义:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteOrg | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| idStr | String[] | | 机构/部门id的字符串数组 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "删除机构成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数idStr删除机构：

① 解析并遍历idStr，每次遍历得到一个id参数，根据参数id查找机构表，根 据查找到的数据判断其parent\_org字段是否为空，即判断其是否有父机构：

1） 若有父机构，修改父级机构数据child\_org字段，然后修改该数据的 isdelete字段，将该字段值改为“1”即可。

2） 若没有父机构则直接根据然后修改该数据的 isdelete字段，将该字段 值改为“1”即可。

* + 1. 配置机构角色接口（机构前端、后台角色不同表，做成同一个接口，传一个判断参数）

**a.接口url:**

/pub/admin/user/organization/setOrgRole.do

**b.接口说明:**

该接口用于给机构配置角色（前端、后台共用）

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| setOrgRole | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构ID | |
| ridStr | String [] | | 遍历页面勾选的角色id拼接的字符串数组 | |
| type | int | | 0=前端角色，1=后台角色 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "配置机构角色成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数type，判断配置的是前端角色、后台角色。调用
2. 根据参数orgid ，ridStr 配置机构角色：

① 根据参数orgid，和每次遍历ridStr数组得到的rid在机构角色表中增加一条机构角色的数据。（需判断数据是否已经存在）

* + 1. 删除机构角色接口（机构前端、后台角色不同表，做成同一个接口，传一个判断参数）

**a.接口url:**

/pub/admin/user/organization/deleteOrgRole.do

**b.接口说明:**

该接口用于删除机构配置的角色用（前端、后台共用）

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteOrgRole | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构ID | |
| ridStr | String [] | | 勾选的角色id的字符串数组 | |
| type | int | | 0=前端角色，1=后台角色 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "删除机构角色成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数type，判断删除的是前端角色、后台角色。调用
2. 根据参数orgid，ridStr 删除机构配置的角色：

① 遍历ridStr中所有的角色rid;

② 根据遍历出来的rid和参数orgid删除机构角色表中对应的数据;

* + 1. 查找已配置的机构或部门角色列表 (前后台共用) （机构前端、后台角色不同表，做成同一个接口，传一个判断参数）

**a.接口url:**

/pub/admin/user/organization/findOrgRole.do

**b.接口说明:**

该接口用于查找机构、部门已配置的角色（前端、后台共用）

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findOrgRole | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构ID | |
| values | String | | 角色名称 | |
| page | int | | 页数 | |
| num | Int | | 每页显示信息数 | |
| rid | String | | 角色id或父角色id | |
| type | Int | | 0：前端 1：后端 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "配置机构角色成功" //返回的信息；

}

}

**e.功能实现说明**

1. 角色id不为空时，查子角色，不分页
   * 1. 查找未配置的机构或部门角色列表 (前后台共用) （感觉没用）

**a.接口url:**

/pub/admin/user/organization/findUnOrgRole.do

**b.接口说明:**

该接口用于给机构配置角色时，查询未配置角色列表

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findUnOrgRole | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构ID | |
| values | String | | 角色名称 | |
| page | int | | 页数 | |
| num | Int | | 每页显示信息数 | |
| rid | String | | 角色id或父角色id | |
| type | Int | | 0：前段 1：后端 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "配置机构角色成功" //返回的信息；

}

}

**e.功能实现说明**

1. 角色id不为空时，查子角色，不分页
   * 1. 查找机构/部门下员工列表接口（前端人员）

**a.接口url:**

pub/admin/user/organization/findOrgStaffList.do

**b.接口说明:**

该接口用于根据条件查询机构下员工列表（前端人员）

**c.接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findOrgStaffList | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| orgid | String | 机构ID |
| pageSize | int | 每页显示信息数 |
| nowpage | Int | 当前页 |
| Keys[] | String | 键 |
| Values[] | String | 值 |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

"id": "8128", //员工ID;

"title": "小明", //员工姓名;

"orgid": "子杰信息", //所属机构;

"account": "221231 ", //账号;

"password": "12", //密码;

"idNumber": "123456", //身份证号;

"rTime": "20", //创建时间;

"rorder": "1" //显示顺序;

},

{ }

]

}

**e.功能实现说明**

1. rid(角色Id)为空的时候：查找机构下所有的员工
   1. 根据orgid, account,peoplename拼接查询语句。把得到的数据list放到infor对象的infos字段中。把list长度放到infor对象的total字段中。
2. rid不为空的时候：查找机构角色下的员工
   1. 根据rid,orgid, account,peoplename拼接查询语句。把得到的数据list放到infor对象的infos字段中。把list长度放到infor对象的total字段中。
3. 设置page对象的其他字段。设置result对象的字段，都放到infor对象中。
4. 返回infor对象。
   * 1. 查找前端员工信息接口 +

**a.接口url:**

/pub/admin/user/organization/findStaffById.do

**b.接口说明:**

该接口用于根据前端员工ID查询员工信息

**c.接口定义:**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findOrgStaffList | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| pid | String | 员工ID |

**d.返回值示例与说明**

{

“infos”:

{

"id": "8128", //员工ID;

"title": "小明", //员工姓名;

"orgid": "子杰信息", //所属机构;

"account": "221231 ", //账号;

"password": "12", //密码;

"idNumber": "123456", //身份证号;

"rTime": "20", //创建时间;

"rorder": "1" //显示顺序;

}

}

**e.功能实现说明**

(1) 根据参数pid查询员工表中id字段值为参数pid值的数据。

* + 1. 查找未配置的部门人员列表（感觉没用）

**a.接口url:**

/pub/admin/user/organization/ findOrgUnStaffList.do

**b.接口说明:**

该接口用于部门未配置的人员列表。

**c.接口定义:**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findOrgUnStaffList | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| orgid | String | 机构ID |
| pageSize | int | 每页显示信息数 |
| nowpage | Int | 当前页 |
| Keys[] | String | 键 |
| Values[] | String | 值 |

**d.返回值示例与说明**

{

“infos”:

{

"id": "8128", //员工ID;

"title": "小明", //员工姓名;

"orgid": "子杰信息", //所属机构;

"account": "221231 ", //账号;

"password": "12", //密码;

"idNumber": "123456", //身份证号;

"rTime": "20", //创建时间;

"rorder": "1" //显示顺序;

}

}

**e.功能实现说明**

(1) 根据参数pid查询员工表中id字段值为参数pid值的数据。

* + 1. 删除员工接口（前端人员）

**a.接口url:**

/pub/admin/user/organization/deleteOrgStaff.do

**b.接口说明:**

该接口用于删除机构下员工

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteOrgStaff | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pid | String | | 员工ID | |
| orgid | String | | 机构或部门id | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "删除员工成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数pid删除员工：
2. 如果是部门人员，则删除【部门人员表】中的数据，如果是机构人员，则删除【机构、部门人员表】和【人员基础表】中的所有相关信息
   * 1. 删除员工角色接口（前端、后台共用）

**a.接口url:**

/pub/admin/user/organization/deleteStaffRole.do

**b.接口说明:**

该接口用于删除员工配置的角色

**c.接口定义:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteStaffRole | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| ridStr | String[] | | 勾选的角色id拼接的字符串数组 | |
| pid | String | | 员工ID | |
| oid | String | | 机构id集合 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "删除员工角色成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数ridStr ，pid，oid删除员工配置的角色，遍历ridStr数组得rid：

① 根据rid和pid删除机构员工角色表中rid为当前遍历得到的rid的值并且peid为参数pid值的数据，oid为参数oid值的数据。

* + 1. 配置员工角色接口（前端、后台共用，传一个判断参数是否是管理员角色）

**a.接口url:**

/pub/admin/user/organization/setStaffRole.do

**b.接口说明:**

该接口用于给员工配置角色用

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| setStaffRole | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pid | String | | 员工ID | |
| ridStr | String[] | | 遍历页面勾选的角色id的字符串数组 | |
| oid | String | | 机构id集合 | |
| type | int | | 0=前端角色，1=后台角色 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "配置员工角色成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数pid,ridStr，oid配置员工角色，遍历ridStr数组，每次遍历得rid，

① 生成UUID，根据参数pid和参数rid，UUID在机构员工角色表中插入一条数据，该数据的id字段为生成的UUID，rid为遍历得到的参数rid参数值，peid字段值为参数pid的值，oid为参数oid值的数据;

* + 1. 查看人员机构/部门前端角色列表

**a.接口url:**

/pub/admin/user/organization/findOrgByrid.do

**b.接口说明:**

~~查看该角色配置给了哪些部门~~。查看该员工在机构/部门配置的角色列表

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findOrgByrid | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| ~~rid~~ | ~~String~~ | | ~~角色~~ | |
| pid | String | | 员工id | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "配置员工角色成功" //返回的信息；

}，

"infos": [ //信息;

{

"id": "80", //机构id;

"title": "部门名称", //部门名称;

},

{

"id": "80", //机构id;

"title": "部门名称", //部门名称;

}

}

**e.功能实现说明**

~~根据角色id，查询该角色所属部门，~~

* + 1. 查看人员机构/部门后台角色列表
    2. 删除员工标签接口

**a.接口url:**

/pub/admin/user/organization/deleteStaffTag.do

**b.接口说明:**

该接口用于删除员工配置的标签

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteStaffTag | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pid | String | | 员工ID | |
| tidStr | String[] | | 勾选的标签id拼接的字符串数组 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "删除员工标签成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据pid，tidStr删除员工配置的标签：

① 遍历tidStr数组,每次遍历得tid,根据pid查找人员表数据，修改该数据中的bid字段值，将该字段中的“tid,”片段用“”替换掉。

* + 1. 配置员工标签接口

**a.接口url:**

/pub/admin/user/organization/setStaffTag.do

**b.接口说明:**

该接口用于给员工配置标签

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| setStaffTag | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pid | String | | 员工ID | |
| tidStr | String[] | | 遍历页面勾选的标签id拼接的字符串数组 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "配置员工标签成功" //返回的信息；

}

}

**e.功能实现说明**

1. 配置员工标签,遍历tidStr数组,每次遍历得tid：

①根据pid查找人员表数据，得到数据中的bid，将tid拼接到bid中，拼接完成后修改人员表中对应数据的bid字段，修改后的值为拼接得到的bid值。

* + 1. 删除机构数据词典

**a.接口url:**

/pub/admin/user/organization/deleteOrgDataDic.do

**b.接口说明:**

该接口用于删除机构配置的数据词典

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteOrgDataDic | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| ids | String[] | | 机构数据词典id数组 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "删除机构数据词典成功" //返回的信息；

}

}

**e.功能实现说明**

1. 遍历didStr数组，每次遍历得did1

① 然后根据orgid，did1删除机构数据词典表中oid字段值为参数orgid值并且 did字段值为遍历得到的参数did1值的数据。

* + 1. 配置机构数据词典

**a.接口url:**

/pub/admin/user/organization/setOrgDataDic.do

**b.接口说明:**

该接口用于给机构配置数据词典用

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| setOrgDataDic | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构ID | |
| didStr | String[] | | 数据词典ID拼接的字符串数组 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "配置机构数据词典成功" //返回的信息；

}

}

**e.功能实现说明**

（1）根据参数orgid，didStr为机构配置数据词典，遍历数组didStr，每次遍历得did1：

① 根据参数orgid，did1,以及生成一个UUID在机构数据词典表插入数据。

* + 1. 导入机构数据词典
    2. 导出机构数据词典
    3. 更新机构数据词典
    4. 查询机构未配置的数据词典列表（感觉没用）

**a.接口url:**

/pub/admin/user/organization/FindUnOrgDataDicList.do

**b.接口说明:**

该接口用于查询未配置的数据词典列表

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| FindUnOrgDataDicList | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构ID | |
| didStr | String[] | | 数据词典ID拼接的字符串数组 | |
| pageSize | int | | 每页显示信息数 | |
| nowpage | Int | | 当前页 | |
| key | String | | 关键字 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "配置机构数据词典成功" //返回的信息；

}

}

**e.功能实现说明**

（1）根据参数orgid，didStr为机构配置数据词典，遍历数组didStr，每次遍历得did1：

① 根据参数orgid，did1,以及生成一个UUID在机构数据词典表插入数据。

* + 1. 查询机构已配置的数据词典列表

**a.接口url:**

/pub/admin/user/organization/ FindOrgDataDicList.do

**b.接口说明:**

该接口用于查询已配置的数据词典列表

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| FindOrgDataDicList | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构ID | |
| didStr | String[] | | 数据词典ID拼接的字符串数组 | |
| pageSize | int | | 每页显示信息数 | |
| nowpage | Int | | 当前页 | |
| key | String | | 关键字 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "配置机构数据词典成功" //返回的信息；

}

}

**e.功能实现说明**

（1）根据参数orgid，didStr为机构配置数据词典，遍历数组didStr，每次遍历得did1：

① 根据参数orgid，did1,以及生成一个UUID在机构数据词典表插入数据。

* + 1. 增加/修改机构接口 +

**a.接口url:**

/pub/admin/user/organization/addOrUpdateOrg.do

**b.接口说明:**

该接口用于增加/修改机构

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| addOrUpdateOrg | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgnization | ZjxxOrgnization （对象） | | 机构表类对象，对象详细参数参照文档6.4.1 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "增加机构（部门）成功" //返回的信息；

}

}

**e.功能实现说明**

（1）根据对象参数orgnization 判断对象中id字段值是否为空：

① id值为空时，生成UUID赋值给对象中id字段，然后再判断该对象中parentOrg 字段值是否为空：

1. 若为空则直接将该对象插入机构表中;
2. 若不为空则先修改机构表中id字段值为对象中parentOrg值的数据的 childOrg字段值，修改其值为生成的UUID值，然后再将该对象插入机构表中;

② id值不为空时：

1） 根据参数id查找机构表中id字段值为参数id值的数据，根据该数据中 的linkruleid字段查找规则表中id字段值为linkruleid字段值的数据， 再根据查找规则表得到的数据的rule\_content字段值，得到转换的数据 表以及参数转化规则，然后根据得到的转换的数据表名以及转化后对应的 参数修改信息。

2）判断对象中的parentOrg字段值是否发生了变化：

I. 若发生了改变，根据原数据中的parent\_org查找数据词典表中id字 段值为原数据中的parent\_org值的数据，修改其child\_org值; 再 根据对象中parentOrg的值查找机构表中id字段值为对象中 parentOrg值的数据，修改其child\_org字段值;最后根据 id更新 数据词典表中该数据。

Ii. 若未发生改变，则直接根据id更新机构表中该数据;

* + 1. 增加/修改员工 +

**a.接口url:**

/pub/admin/user/organization/addStaff.do

**b.接口说明:**

该接口用于增加员工

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| addStaff | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| people | ZjxxPeople （对象） | | 人员表类对象，对象详细参数参照文档6.4.2 | |
| pIds | String | | 父机构ID包括本身 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "修改/添加员工成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据对象参数people判断该对象中的id字段是否为空：

① 若id为空，生成一个UUID赋值给该对象的id字段：

1).将该对象插入人员表中;（说明：要首先获得系统参数，看密码是一次加密还是二次加密，在把密码进行处理加密，放到数据库中）

2).生成一个新的UUID，对象中的orgid字段值，id字段值，rid字段值,分 别整合到机构员工角色表类对象中，然后将该对象插入机构员工角色表中，把pIds加到机构角色员工表中。

② 若id不为空:

1) 根据参数id查找员工表中id字段值为参数id值的数据，~~根据该数据中 的linkruleid字段查找 规则表中id字段值为linkruleid字段值的数据， 再根据查找规则表得到的数据的rule\_content字段值，得到转换的数据 表以及参数转化规则，然后根据得到的转换的数据表名以及转化后对应的 参数修改信息。~~

2) 根据对象中的id查找员工表得到新数据的orgid，判断对象中 orgid 字段值与查找到的数据的orgid是否发生了变化：

i.若未变化，直接根据id更新人员表中该条数据;

ii.若发生了变化：

A.把pIds修改到机构角色员工表中

B.根据id更新人员表中该条数据;

* + 1. 批量导入员工接口

**a.接口url:**

/pub/admin/user/organization/importStaff.do

**b.接口说明:**

该接口用于从excel批量导入员工到数据库

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| importStaff | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
|  |  | |  | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "批量导入员工成功" //返回的信息；

}

}

**e.功能实现说明**

1. 解析上传的excel文件，然后批量添加信息：

① 添加员工信息到人员表中；

1. 在机构员工表中也要添加对应的信息。
   * 1. 批量导出员工接口

**a.接口url:**

/pub/admin/user/organization/exportStaff.do

**b.接口说明:**

该接口用于批量导出员工到excel

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| exportStaff | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构ID | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "批量导出员工成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数orgid批量导出员工到excel：

① 根据orgid查找人员表中orgid字段值为参数orgid值的数据;

1. 将产找到的数据生成excel文件。
   * 1. 机构/部门信息转化接口
     2. 查看员工前端/后台登录记录接口
   1. 数据词典管理
      1. 查找数据词典列表接口

**a.接口url:**

/pub/admin/user/dataDic/findDatadicList.do

**b.接口说明:**

该接口用于根据参数条件查询数据词典列表

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findDatadicList | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pageSize | int | | 每页显示信息数 | |
| nowpage | Int | | 当前页 | |
| dataname | String | | 搜索数据词典名称 | |
| pid | String | | 父级数据词典ID | |
| orgid | String | | 机构ID | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

"id": "80", //数据词典ID;

"title": "数据词典", //数据词典名称;

"description ": "数据词典", //数据词典描述;

"pid": "", //父级数据词典ID;

"cid": "17db2a", //子级数据词典ID集合;

"order\_num": 0 //数据词典顺序;

},

{}

]

}

**e.功能实现说明**

1. 根据参数dataname,pid,orgid查找数据词典列表：

① 当orgid不为空时拼接查找条件，查找机构数据词典表中oid字段值为参数orgid字段值的数据，再根据该数据的did字段值查找数据词典表。

② 当pid不为空时拼接查找条件，查找数据词典表中pid字段值为参数pid的数据;

③ 当pid为空时拼接查找条件，查找数据词典表中pid字段值为空的数据;

④ 当dataname不为空时拼接查找条件，查找数据词典表中title字段值like参数dataname值的数据;

⑤ 根据参数nowpage，pageSize计算出查找条件，追加查找条件limit;

⑥ 根据拼接的sql语句执行查询操作，返回查询列表，并根据返回的数据列表，分页实体类Page,返回信息类Info整合数据。

注：page对象字段详细情况参照6.6.1分页类对象，info对象的字段详细信息参照6.6.2返回信息类对象。

* + 1. 查找数据词典信息接口

**a.接口url:**

/pub/admin/user/dataDic/findDatadicById.do

**b.接口说明:**

该接口用于根据参数条件查询数据词典列表

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findDatadicById | | ZjxxDataDictionary | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| did | String | | 数据词典ID | |

**d.返回值示例与说明**

{infos：[ {

"id": "80", //菜单ID;

"title": "机构A菜单", //菜单名称;

"description ": "菜单", //菜单描述;

"isFinal": "1", //是否终极菜单;

"parentMenu": "", //父菜单ID;

"childMenu": "17db2a", //子菜单ID集合;

"pageUrl": "", //页面URL;

"menuOrder": 0 //菜单顺序;

"whitelist": 0 //白名单;

"blacklist": 0 //黑名单;

"seegroupid": 0 //可浏览用户;

}

]

}

**e.功能实现说明**

1. 根据参数menuid查找菜单信息，menuid参数不能为空:查找菜单表中id字段值为参数menuid值的数据;

**e.功能实现说明**

（1）根据参数did查询数据词典表中id字段值为参数did字段值的数据。

* + 1. 删除数据词典接口

**a.接口url:**

/pub/admin/user/dataDic/deleteDatadic.do

**b.接口说明:**

该接口用于删除数据词典

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteDatadic | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| idStr | String[] | | 数据词典ID拼接的字符串数组 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "删除数据词典成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数idStr删除数据词典，遍历idStr数组，每次遍历得参数id，根据参 数id查找数据词典表中id字段值为参数id值的数据：

① 根据id查找数据词典表中id字段值为参数id值的数据，根据该数据中的 linkruleid字段查找规则表中id字段值为linkruleid字段值的数据，再根据查 找规则表得到的数据的rule\_content字段值，得到转换的数据表以及参数id对应 的转换后的信息id，然后根据得到的转换的数据表名以及id删除该表中该数据.

② 判断该数据是否有父数据词典：

1） 若有父数据词典则修改父级数据词典的词典cid字段，然后删除该数据词 典的数据记录以及递归查找并删除子级数据词典;

2） 若无父数据词典直接删除该数据词典的数据记录以及递归查找并删除子级 数据词典;

3） 删除每个数据词典数据记录时都要删除机构数据词典表中did字段值为删 除的数据词典id值的数据;

* + 1. 修改数据词典顺序接口

**a.接口url:**

/pub/admin/user/dataDic/updateDataDicOrder.do

**b.接口说明:**

该接口用于修改数据词典顺序

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| updateDataDicOrder | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| idStr | String[] | | 遍历页面数据词典ID拼接的字符串数组 | |
| orderStr | String[] | | 遍历页面数据词典顺序拼接的字符串数组 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "修改数据词典顺序成功" //返回的信息；

}

}

**e.功能实现说明**

（1）根据idStr，orderStr修改数据词典顺序,遍历idStr，orderStr数组，每次遍历得到id,order,根据参数id，order修改数据词典表中id字段值为参数id值的数据的order\_num字段，修改其值为参数order的值。

* + 1. 导入数据词典到excel接口
    2. 导出数据词典到excel接口

**a.接口url:**

/pub/admin/user/dataDic/exportDatadic.do

**b.接口说明:**

该接口用于导出数据词典到excel

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| exportDatadic | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
|  |  | |  | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "导出数据词典成功" //返回的信息；

}

}

**e.功能实现说明**

（1）查询数据词典表，查找所有的数据词典数据，并且按照父子关系的结构导出数据 词典到excel表中。

* + 1. 生成Json接口

**a.接口url:**

/pub/admin/user/dataDic/dataDicToJson.do

**b.接口说明:**

该接口用于将数据词典生成Json

**c.接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| dataDicToJson | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 数据词典id |

**d.返回值示例与说明**

{

“infos”:

[{

"id": "80", //数据词典ID;

"title": "数据词典", //数据词典名称;

"description ": "数据词典", //数据词典描述;

"pid": "", //父级数据词典ID;

"cid": "17db2a", //子级数据词典ID集合;

"order\_num": 0 //数据词典顺序;

},

{

"id": "80", //数据词典ID;

"title": "数据词典", //数据词典名称;

"description ": "数据词典", //数据词典描述;

"pid": "", //父级数据词典ID;

"cid": "17db2a", //子级数据词典ID集合;

"order\_num": 0 //数据词典顺序;

},

{ }

]

}

**e.功能实现说明**

（1）根据参数id查找数据词典表中id字段值为参数id值的数据，以及其子孙数据，返回json对象数组

* + 1. 增加/修改数据词典接口

**a.接口url:**

/pub/admin/user/dataDic/addOrUpdateDataDic.do

**b.接口说明:**

该接口用于增加/修改数据词典

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| addOrUpdateDataDic | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| dictionary | ZjxxDataDictionary (对象) | | 数据词典对象，对象详细参数参照文档6.4.5 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "增加数据词典成功" //返回的信息；

}

}

或

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "修改数据词典成功" //返回的信息；

}

}

**e.功能实现说明**

（1）根据对象参数dictionary判断对象中id字段值是否为空：

① id值为空时，生成UUID赋值给对象中id字段，然后再判断该对象中pid字段 值是否为空：

1）若为空则直接将该对象插入数据词典表中;

2）若不为空则先修改数据词典表中id字段值为对象中pid值的数据的cid 字段值，修改其值为生成的UUID值，然后再将该对象插入数据词典表中;

② id值不为空：

1）根据参数id查找数据词典表中id字段值为参数id值的数据，根据该数据 中的linkruleid字段查找规则表中id字段值为linkruleid字段值的数据， 再根据查找规则表得到的数据的rule\_content字段值，得到转换的数据 表以及参数转化规则，然后根据得到的转换的数据表名以及转化后对应的 参数修改信息。

2）判断对象中的pid字段值是否发生了变化：

i. 若发生了改变，根据原数据中的pid查找数据词典表中id字段值为 原数据中的pid值的数据，修改其cid值; 再根据对象中pid的值查 找数据词典表中id字段值为对象中pid值的数据，修改其cid字段 值;最后根据id更新数据词典表中该数据。

Ii .若未发生改变，则直接根据id更新数据词典表中该数据;

* + 1. 更新数据词典接口
  1. 角色管理
     1. 查找角色列表接口

**a.接口url:**

/pub/admin/user/role/findRoleList.do

**b.接口说明:**

该接口用于根据参数条件查找角色列表

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findRoleList | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pageSize | int | | 每页显示信息数 | |
| nowpage | Int | | 当前页 | |
| rolename | String | | 输入的角色名称 | |
| pid | String | | 父角色ID | |
| orgid | String | | 机构ID | |
| peopleid | String | | 人员ID | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

"id": "wedff887sds", //角色ID;

"title": "角色11", //角色名称;

"pid": "123", //父角色ID;

"rorder": 0, //角色顺序;

"description": "123", //角色描述;

"isdelete": "123", //是否删除;

"cid": "123", //子角色ID集合;

"orgid": "123" //

},

{}

]

}

**e.功能实现说明**

1. 根据参数rolename，pid，orgid，peopleid查询角色：

① 当rolename不为空时拼接查找条件，查找角色表中title字段值like参数 rolename值的数据;

② 当orgid不为空时拼接查找条件，查找机构角色表中oid字段值为参数orgid 值的数据，根据查找到的数据的rid字段值查找角色表;

③ 当peopleid不为空时拼接查找条件，查找机构员工角色表中peid字段值为参 数peopleid值的数据，根据查找到的数据的rid字段值查找角色表;

④ 当pid不为空时拼接查找条件，查找角色表中pid字段值为参数pid值的数据;

⑤ 当pid为空时拼接查找条件，查找角色表中pid字段值为空的数据;

⑥ 拼接查找条件，查找角色表中isdelete值为“0”的数据;

⑦ 根据参数nowpage，pageSize计算出查找条件，追加查找条件limit;

⑧ 根据拼接的sql语句执行查询操作，返回查询列表，并根据返回的数据列表， 分页实体类Page,返回信息类Info整合数据。

注：page对象字段详细情况参照6.6.1分页类对象，info对象的字段详细信息参照6.6.2返回信息类对象。

* + 1. 4.3.2 查找角色信息接口

**a.接口url:**

/pub/admin/user/role/findRole.do

**b.接口说明:**

该接口用于查找角色信息

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findRoleList | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| rid | String | | 角色ID | |

**d.返回值示例与说明**

{

“infos”:

{

"id": "wedff887sds", //角色ID;

"title": "角色11", //角色名称;

"pid": "123", //父角色ID;

"rorder": 0, //角色顺序;

"description": "123", //角色描述;

"isdelete": "123", //是否删除;

"cid": "123", //子角色ID集合;

"orgid": "123" //机构ID

}

}

**e.功能实现说明**

（1）根据参数rid查找角色表中id字段值为rid值的数据。

* + 1. 4.3.3 删除角色接口

**a.接口url:**

/pub/admin/user/role/deleteRole.do

**b.接口说明:**

该接口用于删除角色

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteRole | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| idStr | String | | 页面勾选的角色ID拼接的字符串，多个逗号隔开，例如：id1,id2,id3, | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "删除角色成功" //返回的信息；

}

}

**e.功能实现说明**

1. 删除角色，遍历idStr数组，每次遍历得参数id，

① 根据id查找角色表中id字段值为参数id值的数据，根据该数据中的linkruleid字段查找规则表中id字段值为linkruleid字段值的数据，再根据查找规则表得到的数据的rule\_content字段值，得到转换的数据表以及参数id对应的转换后的信息id，然后根据得到的转换的数据表名以及id删除该表中该数据

② 根据参数id查找角色表中id字段值为参数id值的数据，判断该数据是否有父 角色：

1） 若有父角色则修改父角色的cid字段，然后修改该角色的isdelete字段 修改为“1”;

2） 若没有父角色则直接修改该角色的isdelete字段，修改其值为“1”。

* + 1. 4.3.4 修改角色顺序接口

**a.接口url:**

/pub/admin/user/role/updateRoleOrder.do

**b.接口说明:**

该接口用于修改角色顺序

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| updateRoleOrder | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| idStr | String[] | | 遍历页面角色ID拼接的字符串数组 | |
| orderStr | String[] | | 遍历页面角色顺序拼接的字符串数组 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "修改角色顺序成功" //返回的信息；

}

}

**e.功能实现说明**

（1）根据参数idStr，orderStr修改角色顺序：遍历idStr，orderStr，每次遍历得到id,order,根据参数id，order修改角色表中id字段值为参数id值的数据的rorder字段，修改其值为参数order的值。

* + 1. 4.3.5 增加/修改角色接口

**a.接口url:**

/pub/admin/user/role/addRole.do

**b.接口说明:**

该接口用于增加/修改角色

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| addRole | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| role | ZjxxRole （对象） | | 角色表类对象，对象详细参数参照文档6.4.3 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "增加角色成功" //返回的信息；

}

}

或

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "修改角色成功" //返回的信息；

}

}

**e.功能实现说明**

（1）根据对象参数role判断对象中id字段值是否为空：

①id值为空时，生成UUID赋值给对象中id字段，然后再判断该对象中pid字段 值是否为空：

1）若为空则直接将该对象插入角色表中;

2）若不为空则先修改角色表中id字段值为对象中pid值的数据的cid字段 值，修改其值为生成的UUID值，然后再将该对象插入角色表中;

② id值不为空时：

1）根据参数id查找数据词典表中id字段值为参数id值的数据，根据该数据 中的linkruleid字段查找规则表中id字段值为linkruleid字段值的数据， 再根据查找规则表得到的数据的rule\_content字段值，得到转换的数据 表以及参数转化规则，然后根据得到的转换的数据表名以及转化后对应的 参数修改信息

2）判断对象中的pid字段值是否发生了变化：

i.若发生了改变，根据原数据中的pid查找角色表中id字段值为原数据中的pid值的数据，修改其cid值; 再根据对象中pid的值查找角色表中id字段值为对象中pid值的数据，修改其cid字段值;最后根据id更新角色表中该数据。

ii.若未发生改变，则直接根据id更新角色表中该数据;

* + 1. 4.3.6 导入角色接口
    2. 4.3.7 导出角色接口
  1. 菜单管理
     1. **查找菜单信息接口**

**a.接口url:**

/pub/admin/user/menu/findMenu.do

**b.接口说明:**

接该接口用于查找菜单信息

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findMenu | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| menuid | String | | 菜单ID | |

**d.返回值示例与说明**

{

“infos”:

{

"id": "80", //菜单ID;

"title": "机构A菜单", //菜单名称;

"description ": "菜单", //菜单描述;

"isFinal": "1", //是否终极菜单;

"pid": "", //父菜单ID;

"child": "17db2a", //子菜单ID集合;

"shortTitle": "17db2a", //菜单名称简称;

"englishTitle": "17db2a", //菜单名称英文;

"orgid": "kjflkfj", //机构id;

"type": "1", //类型;

}，

“result”：{

“flag”：1， //成功标志

“msg”：“查找成功” //说明

}

}

**e.功能实现说明**

1. 根据菜单id来查找菜单数据。
2. 把找到的数据放到infor对象的info字段中。再给infor对象设置结果集result。
   * 1. **查找机构或部门前端/后台菜单列表**

**a.接口url:**

/pub/admin/user/menu/findMenuList.do

**b.接口说明:**

接该接口用于根据参数条件查询菜单

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findMenuList | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构id | |
| type | int | | 0:前段 1：后端 | |
| Menuid | String | | 菜单id | |

**d.返回值示例与说明**

{

“infos”:

[{

"id": "80", //菜单ID;

"title": "机构A菜单", //菜单名称;

"description ": "菜单", //菜单描述;

"isFinal": "1", //是否终极菜单;

"pid": "", //父菜单ID;

"child": "17db2a", //子菜单ID集合;

"shortTitle": "17db2a", //菜单名称简称;

"englishTitle": "17db2a", //菜单名称英文;

"orgid": "kjflkfj", //机构id;

"type": "1", //类型;

}，{…}],

“result”：{

“flag”：1， //成功标志

“msg”：“查找成功” //说明

}

}

**e.功能实现说明**

1. 把参数orgid（机构id）和pid（父菜单）放到map中
2. 调用dao层方法，传入map参数，把得到数据放到infor对象中的info字段中。
3. 在给对象设置result字段的数据。
4. 返回infor对象。
   * 1. 新增/修改**机构前端/后台菜单**
     2. 新增**部门前端/后台菜单**
     3. 删除**机构前端/后台菜单**
     4. 删除**部门前端/后台菜单**

**a.接口url:**

/pub/admin/user/menu/deleteMenu.do

**b.接口说明:**

该接口用于删除菜单

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteMenu | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| id | String | | 菜单Id | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "删除菜单成功" //返回的信息；

}

}

**e.功能实现说明**

1. 判断该菜单有没有被配置给部门、角色、部门角色、人员，若有，则进行提示。若无，则删除菜单表和机构表中的菜单集合字段
   * 1. 修改**机构前端/后台菜单顺序**
     2. **查找员工前端菜单列表**
     3. **新增员工前端菜单列表**
     4. **删除员工前端菜单列表**
     5. **查找员工后台菜单列表**
     6. **新增员工后台菜单列表**
     7. **删除员工后台菜单列表**
     8. **查找机构或部门前端角色菜单列表**

**a.接口url:**

/pub/admin/user/menu/findMenuList.do

**b.接口说明:**

接该接口用于根据参数条件查询菜单

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findMenuList | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构id | |
| type | int | | 0:前段 1：后端 | |
| Menuid | String | | 菜单id | |
| jid | String | | 角色 | |

**d.返回值示例与说明**

{

“infos”:

[{

"id": "80", //菜单ID;

"title": "机构A菜单", //菜单名称;

"description ": "菜单", //菜单描述;

"isFinal": "1", //是否终极菜单;

"pid": "", //父菜单ID;

"child": "17db2a", //子菜单ID集合;

"shortTitle": "17db2a", //菜单名称简称;

"englishTitle": "17db2a", //菜单名称英文;

"orgid": "kjflkfj", //机构id;

"type": "1", //类型;

}，{…}],

“result”：{

“flag”：1， //成功标志

“msg”：“查找成功” //说明

}

}

**e.功能实现说明**

1. 把参数orgid（机构id）和pid（父菜单）放到map中
2. 调用dao层方法，传入map参数，把得到数据放到infor对象中的info字段中。
3. 在给对象设置result字段的数据。
4. 返回infor对象。
   * 1. **新增机构或部门前端角色菜单列表**
     2. **删除机构或部门前端角色菜单列表**
     3. **查找机构或部门后台角色菜单列表**
     4. **删除机构或部门后台角色菜单列表**
     5. **新增机构或部门后台角色菜单列表**
     6. **查找人员前端菜单列表**

**a.接口url:**

/pub/admin/user/menu/findMenuList.do

**b.接口说明:**

接该接口用于根据参数条件查询菜单

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findMenuList | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| orgid | String | | 机构id | |
| type | int | | 0:前段 1：后端 | |
| Menuid | String | | 菜单id | |
| Peoid | String | | 人员id | |

**d.返回值示例与说明**

{

“infos”:

[{

"id": "80", //菜单ID;

"title": "机构A菜单", //菜单名称;

"description ": "菜单", //菜单描述;

"isFinal": "1", //是否终极菜单;

"pid": "", //父菜单ID;

"child": "17db2a", //子菜单ID集合;

"shortTitle": "17db2a", //菜单名称简称;

"englishTitle": "17db2a", //菜单名称英文;

"orgid": "kjflkfj", //机构id;

"type": "1", //类型;

}，{…}],

“result”：{

“flag”：1， //成功标志

“msg”：“查找成功” //说明

}

}

**e.功能实现说明**

1. 把参数orgid（机构id）和pid（父菜单）放到map中
2. 调用dao层方法，传入map参数，把得到数据放到infor对象中的info字段中。
3. 在给对象设置result字段的数据。
4. 返回infor对象。
   * 1. **查询人员前端菜单及按钮接口**
     2. **查询人员前端私有菜单及按钮接口**
     3. **新增人员前端私有菜单及按钮接口**
     4. **删除人员前端私有菜单及按钮接口**
     5. **查询人员后台私有菜单及按钮接口**
     6. **新增人员后台私有菜单及按钮接口**
     7. **删除人员后台私有菜单及按钮接口**
     8. **查找机构后台按钮接口**
     9. **新增/修改机构后台按钮接口**
     10. **删除机构后台按钮接口**
     11. **查找功能列表接口**

**a.接口url:**

/pub/admin/user/menu/findFunctionList.do

**b.接口说明:**

接该接口用于根据参数条件查询菜单

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findFunctionList | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| menuid | String | | 菜单id | |

**d.返回值示例与说明**

{

“infos”:

[{

"id": "80", //菜单功能ID;

"title": "机构A菜单", //功能名称;

"description ": "菜单", //功能描述;

"shortTitle": "17db2a", //功能名称简称;

"englishTitle": "17db2a", //功能名称英文;

"menuid": "kjflkfj", //菜单id;

"picture": "1", //图片;

"url": "1", //页面url;

"page": "1", //页面;

}，{…}],

“result”：{

“flag”：1， //成功标志

“msg”：“查找成功” //说明

}

}

**e.功能实现说明**

1. 把参数menuid（菜单id）放到map中
2. 调用dao层方法，传入map参数，把得到数据放到infor对象中的info字段中。
3. 在给对象设置result字段的数据。
4. 返回infor对象。
   * 1. **查找菜单功能信息接口**

**a.接口url:**

/pub/admin/user/menu/findFunction.do

**b.接口说明:**

接该接口用于查找菜单功能信息

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findFunction | | Infor | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| id | String | | 菜单功能ID | |

**d.返回值示例与说明**

{

“infos”:

[{

"id": "80", //菜单功能ID;

"title": "机构A菜单", //功能名称;

"description ": "菜单", //功能描述;

"shortTitle": "17db2a", //功能名称简称;

"englishTitle": "17db2a", //功能名称英文;

"menuid": "kjflkfj", //菜单id;

"picture": "1", //图片;

"url": "1", //页面url;

"page": "1", //页面;

}]，

“result”：{

“flag”：1， //成功标志

“msg”：“查找成功” //说明

}

}

**e.功能实现说明**

把菜单id存到不同的表里，根据type区分表

* + 1. **删除菜单接口**

**a.接口url:**

/pub/admin/user/menu/removeMenu.do

**b.接口说明:**

该接口用于增加菜单功能

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| removeMenu | | Result | | Json |
| 参数说明： | |
| 参数名称 | 参数类型 | | 备注 | |
| sid | String | | 部门id或部门角色id或人员id或机构角色id | |
| type | int | | 0-部门，1-部门或机构角色，2-人员， | |
| ids | String[] | | 菜单id | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "成功" //返回的信息；

}

}

**e.功能实现说明**

1. 把菜单id从不同的表中移除，根据type区分表
   1. 标签管理
      1. 查找标签列表接口

**a.接口url:**

/pub/admin/user/menu/findTagList.do

**b.接口说明:**

该接口用于根据参数条件查询标签列表

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findTagList | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| pageSize | int | | 每页显示 | |
| nowpage | Integer | | 当前页 | |
| tagname | String | | 输入的菜单名称 | |
| pid | String | | 员工ID | |

**d.返回值示例与说明**

{

"page": { //分页信息;

"page": 2, //当前页;

"pages": 10, //总页数;

"lines": 100, //信息总条数;

"pageSize": 10, //每页显示信息数;

"pageStart": 1 //起始页;

},

"total": 8, //当前页信息条数;

"infos": [ //信息;

{

"id": "wedff887sds", //标签ID；

"title": "标签1", //标签名称;

"rorder": 0, //标签顺序;

"description": "123" //标签描述;

},

{ }

]

}

**e.功能实现说明**

1. 根据参数tagname，pid查询标签列表：

① 当pid不为空时拼接查找条件，查找人员表中id字段值为参数pid值的数据的bids字段，解析并遍历bid得bid,再根据每次遍历得到的参数bid查找标签表;

② 当tagname不为空时拼接查找条件，查找标签表中title字段值like参数tagname值的数据;

③ 根据参数nowpage，pageSize计算出查找条件，追加查找条件limit;

④ 根据拼接的sql语句执行查询操作，返回查询列表，并根据返回的数据列表，分页实体类Page,返回信息类Info整合数据。

注：page对象字段详细情况参照6.6.1分页类对象，info对象的字段详细信息参照6.6.2返回信息类对象。

* + 1. 查找标签信息接口

**a.接口url:**

/pub/admin/user/menu/findTagById.do

**b.接口说明:**

该接口用于查询标签信息

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| findTagById | | Info | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| tagid | String | | 标签ID | |

**d.返回值示例与说明**

{

"id": "wedff887sds", //标签ID；

"title": "标签1", //标签名称;

"rorder": 0, //标签顺序;

"description": "123" //标签描述;

}

**e.功能实现说明**

1. 根据参数tagid查找标签表中id字段值为参数tagid值的数据。
   * 1. 删除标签接口

**a.接口url:**

/pub/admin/user/tag/deleteTag.do

**b.接口说明:**

该接口用于删除标签

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteTag | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| idStr | String[] | | 页面勾选的标签ID拼接的字符串数组 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "删除标签成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据参数idStr删除标签，解析并遍历参数idStr，每次遍历得参数id：

① 根据id查找标签表中id字段值为参数id值的数据，根据该数据中的linkruleid字段查找规则表中id字段值为linkruleid字段值的数据，再根据查找规则表得到的数据的rule\_content字段值，得到转换的数据表以及参数id对应的转换后的信息id，然后根据得到的转换的数据表名以及id删除该表中该数据

② 根据参数id删除标签表中id字段值为参数id值的数据

③ 修改人员表中bid字段。

* + 1. 导入标签

**a.接口url:**

/pub/admin/user/tag/importTag.do

**b.接口说明:**

该接口用于从excel批量导入标签到数据库

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| importTag | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
|  |  | |  | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "批量导入标签成功" //返回的信息；

}

}

**e.功能实现说明**

（1）解析上传的excel文件，然后批量添加信息。

* + 1. 导出标签

**a.接口url:**

/pub/admin/user/tag/exportTag.do

**b.接口说明:**

该接口用于导出标签到excel

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| exportTag | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
|  |  | |  | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "导出标签成功" //返回的信息；

}

}

**e.功能实现说明**

（1）查询标签表，查找所有的标签，并且导出标签到excel表中。

* + 1. 增加/修改标签接口

**a.接口url:**

/pub/admin/user/tag/addOrUpdateTag.do

**b.接口说明:**

该接口用于增加标签用

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| addTag | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| tag | ZjxxTag （对象） | | 标签表类对象,对象详细参数参照文档6.4.6 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "0", //标志，0为成功，1为操作失败

"msg": "增加标签成功" //返回的信息；

}

}

**e.功能实现说明**

(1） 根据对象参数tag判断对象中id字段值是否为空：

① id值为空时，生成UUID赋值给对象中id字段，然后将对象插入标签表中;

② id字段不为空时

1）根据参数id查找标签表中id字段值为参数id值的数据，根据该数据 中的linkruleid字段查找规则表中id字段值为linkruleid字段值的数据， 再根据查找规则表得到的数据的rule\_content字段值，得到转换的数据 表以及参数转化规则，然后根据得到的转换的数据表名以及转化后对应的 参数修改信息;

2） 根据对象中id的值查找标签表中id字段值为对象中id值的数据，对该 数据进行更新，更新的字段及字段值为对象参数中对应的字段和字段值。

* + 1. 修改标签顺序接口

**a.接口url:**

/pub/admin/user/tag/updateTagOrder.do

**b.接口说明:**

该接口用于修改标签顺序

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| updateTagOrder | | Result | | Json |
| 参数说明： | | | | |
| 参数名称 | 参数类型 | | 备注 | |
| idStr | String[] | | 遍历页面角色ID拼接的字符串数组 | |
| orderStr | String[] | | 遍历页面角色顺序拼接的字符串数组 | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "修改标签顺序成功" //返回的信息；

}

}

**e.功能实现说明**

（1）根据参数idStr，orderStr修改标签顺序：遍历idStr，orderStr，每次遍历得到id,order,根据参数id，order修改标签表中id字段值为参数id值的数据的rorder字段，修改其值为参数order的值。

* 1. 按钮管理
     1. 查询该机构角色的按钮配置的值

**a.接口url:**

/pub/admin/findbutton.do

**b.接口说明:**

**查询配置按钮的信息**

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| findbutton | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| userid | String | | 登录cookie中用户ID | | |
| orgid | String | | 机构ID | | |
| rid | String | | 角色ID | | |

**d.返回值示例与说明**{

{

"result": { //结果

"msg": "", //信息 操作信息按钮配置的值

"flag": "1" //标识

}

}

**e.功能实现说明**

（1）判断userid是否为空

（2）若果userid为空值，说明参数肯定传了orgid和rid，根据这两个参数到机构角色表中查询配置项的值，输出即可。

（3）如果userid不是空值，根据userid到管理员表中查询该用户的机构ID和角色ID ，根据这两个参数到机构角色表中查询配置项的值，输出即可。

* + 1. 更新机构角色的按钮配置的值

**a.接口url:**

/pub/admin/addbutton.do

**b.接口说明:**

**查询配置按钮的信息**

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| findbutton | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| buttonstring | String | | 配置项的处理后的值 | | |
| orgid | String | | 机构ID | | |
| rid | String | | 角色ID | | |

**d.返回值示例与说明**{

{

"result": { //结果

"msg": "", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

（1）根据参数orgid和rid，根据这两个参数到机构角色表中查询配置项的值。

（2）如果该值为空，说明已经配置过权限，直接修改即可。

（3）如果不是空值，说明没有配置过，插入一条机构角色的配置项的值。

* + 1. 获取按钮列表

**a.接口url:**

/pub/admin/GetPageButtonList.do

**b.接口说明:**

**查询配置按钮的信息**

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| GetPageButtonList | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| page | String | | 页数 | | |
| orgid | String | | 机构ID | | |
| num | int | | 条数 | | |

**d.返回值示例与说明**{

{

"result": { //结果

"msg": "", //信息

"flag": "1" //标识

}，

"infos": [ //信息;

{

"buttonid": "wedff887sds", //按钮ID；

"buttonname": "按钮1", //按钮名称;

"htmlname": 0, //按妞页面;

},

{

"buttonid": "wedff887sds", //按钮ID；

"buttonname": "按钮1", //按钮名称;

"htmlname": 0, //按妞页面;

}

]

}

**e.功能实现说明**

（1）根据参数orgid，获取按钮列表

* + 1. 获取按钮详情

**a.接口url:**

/pub/admin/GetPageButtonById.do

**b.接口说明:**

**查询配置按钮的信息**

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| GetPageButtonById | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| id | String | | 按钮id | | |

**d.返回值示例与说明**{

{

"result": { //结果

"msg": "", //信息

"flag": "1" //标识

}，

"infos": [ //信息;

{

"buttonid": "wedff887sds", //按钮ID；

"buttonname": "按钮1", //按钮名称;

"htmlname": 0, //按妞页面;

},

]

}

**e.功能实现说明**

（1）根据参数orgid，获取按钮列表

* + 1. 修改和增加按钮

**a.接口url:**

/pub/admin/AddOrUpdataPageButton.do

**b.接口说明:**

**查询配置按钮的信息**

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| GetPageButtonById | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| ZjxxPageButton | obj | | 按钮对象 | | |

**d.返回值示例与说明**{

{

"result": { //结果

"msg": "增加或修改成功！", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

（1）根据参数orgid，获取按钮列表

* + 1. 删除按钮

**a.接口url:**

/pub/admin/DeletePageButton.do

**b.接口说明:**

**查询配置按钮的信息**

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| GetPageButtonById | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| id | String | | 按钮id | | |

**d.返回值示例与说明**{

{

"result": { //结果

"msg": "删除成功！", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

（1）根据参数orgid，获取按钮列表

* 1. 管理员管理
     1. 增加管理员

**a.接口url:**

/pub/admin/AddOrgAdmin.do

1. **接口说明:**

**增加机构机构管理员。**

**c. 接口定义：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接口定义 | | | | | |
| 接口名 | | 对象名 | | | 返回值类型 |
| GetadminList | | Result | | | Json |
| 参数说明： | | | |  | |
| 参数名称 | 参数类型 | | 备注 | | |
| Zjxxadmin | Zjxxadmin | | 管理员对象 | | |
| Orgid | String | | 机构id | | |

**d.返回值示例与说明**{

{

"result": { //结果

"msg": "添加成功！", //信息

"flag": "1" //标识

}

}

**e.功能实现说明**

添加机构管理员。

* + 1. **管理员匹配员工接口**
    2. **员工匹配管理员接口**

**a.接口url:**

/pub/admin/user/organization/persongoup.do

**b.接口说明:**

该接口用于把员工升级为管理员

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| persongoup | | Result | | Json |
| 参数说明： | |  | | |
| 参数名称 | 参数类型 | | 备注 | |
| Id | String | | 员工id | |
| orgid | String | | 机构id | |
| rid | String | | 管理员角色id | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "成功" //返回的信息；

}

}

**e.功能实现说明**

1. 根据人员id查管理员，如果已经升级过，就返回已经升级过。
2. 根据人员id查出人员表一条数据
3. 创建管理员对象，根据查出的数据给管理员对象赋值。
4. 把管理员对象插入到数据库中。
5. 创建Result返回对象，并赋值参数，返回对象。
   * 1. **删除管理员接口**

**a.接口url:**

/pub/admin/user/organization/deleteAdmin.do

**b.接口说明:**

该接口用于删除管理员

**c.接口定义：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口定义 | | | | |
| 接口名 | | 对象名 | | 返回值类型 |
| deleteAdmin | | Result | | Json |
| 参数说明： | |  | | |
| 参数名称 | 参数类型 | | 备注 | |
| Id | String | | 管理员id | |

**d.返回值示例与说明**

{

"result": { //结果集;

"flag": "1", //标志，1为成功，0为操作失败

"msg": "成功" //返回的信息；

}

}

**e.功能实现说明**

* + 1. **修改管理员信息接口**

1. 通用方法
   1. 数据表与系统模型
      1. 生成文件名方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| createFileName | | String | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| Classid  Infoid | | | |

方法逻辑说明：

1. 调用方法（getcontent(title)）取出在系统参数中的“文件编号字段（fileNum,10进制）”的值
2. 调用方法updatecontent(title content)，将fileNum+1重新存入系统参数
3. 将取到的fileNum（10进制）转化为36进制，的文件名filename。
4. 取filename的第1位作为缓存表后缀编号（按照0~9~a~z的规则+1，z后面则回到0），找到对应的缓存表
5. 向缓存表中存入相关信息（infoid（信息id）,classid,filename等）。
6. 把文件名filename写入信息主表
7. 返回文件名。
   * 1. 生成页面信息方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| createPageInfo | | String | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| id | String | | 信息id |
| classid | String | | 栏目id |
| filename | String | | 文件名（唯一，作为主键） |
| cacheInfoTbname | String | | 缓存表分表 |

方法逻辑说明：

1. 根据classid查询信息所在系统模型id以及内容模板id、再根据系统模型id取得数据表名
2. 根据表名和id查找到信息数据执行查询语句
3. 根据内容模板id取得模板内容
4. 根据系统模型id查找到系统模型信息。获取内容字段
5. 遍历内容字段、依次将内容模板中的标签替换为信息数据中的值
6. 调用[固有标签替换方法](#_5.1.6__固有标签替换方法)，调用[特殊标签处理方法](#_5.1.7__特殊标签处理方法)
7. 将替换好的temptext、filename写入缓存表分表cacheInfoTbname。
8. 返回
   * 1. 生成列表信息方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| createListInfo | | int | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| classid | String | | 栏目id |
| page | Int | | 指定页码 |
| filename | String | | 文件名 |
| cacheInfoTbname | String | | 缓存表分表 |

方法逻辑说明：

1. 根据classid查询信息所在系统模型id以及列表模板id、再根据系统模型id取得数据表名
2. 根据列表模板id取得模板内容temptext。
3. 根据系统模型id查找到系统模型信息。获取列表字段
4. 遍历列表字段、组成查询语句需要查询的字段。
5. 如果page为空，则设置page为1。根据栏目id获取每页显示信息数
6. 根据需要查询的字段和page组成sql语句，查询信息列表。
7. 将列表模板中的标签替换为信息列表中的值。
8. 调用[固有标签替换方法](#_5.1.6__固有标签替换方法)，调用[特殊标签处理方法](#_5.1.7__特殊标签处理方法)
9. 在filename后面加上页码page，如果是第一页则省略，第二页开始组成形如filename\_page的新文件名，将替换好的temptext、filename写入缓存表分表cacheInfoTbname。
10. 返回
    * 1. 刷新条件列表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| refreshConditionList | | int | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| classid | String | | 栏目id |

前言：

刷新条件列表本质是重新生成缓存，因为条件列表的排列组合复杂度是几何增长的，无法一个个直接生成页面。

方法逻辑说明：

检查参数classid、为空则返回信息“来源错误”

非空则根据classid获取栏目信息，取出条件集合字段。若为空则返回信息“非条件列表”

* + 1. 生成条件列表方法（单页）

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| createConditionList | | int | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| filename | String | | 文件名 |

方法逻辑说明：

解析文件名、根据条件缓存表取出并组合条件后、组成该列表文件的sql条件、查询后替换标签内容并返回

* + 1. 固有标签替换方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| replaceTag | | String | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| temptext | String | | 模板内容 |

方法逻辑说明：

1. 返回替换后的模板内容
   * 1. 特殊标签处理方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| manageSpecialTag | | String | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| temptext | String | | String |

方法逻辑说明：

1. 返回替换后的模板内容
   * 1. 归档方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| fileWay | | int | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| classid | String | | 栏目id |
| ids | String | | 转入数据表名 |
| start | Int | | 转移的起点下标 |
| size | Int | | 转移多少条数据 |

方法逻辑说明：

1. 检查参数ids、若非空、则拆分后组成sql条件【where id in (‘id1’, ‘id2’, ‘id3’)】根据classid结合id条件（如果有的话）组成sql查询语句
2. 若id为空、则读取classid中归档时间字段。生成sql语句。再根据start、size拼接limit条件
3. 执行查询语句
4. 遍历查询结果、依次把数据内容复制进归档表后删除原信息并记录归档成功次数
5. 返回归档成功次数
   * 1. 信息操作方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| operateInfoWay | | int | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| infojson | String | | 参数集合 |
| taskid | String | | 任务ID |

方法逻辑说明：

1. 从infojson中获取各个参数
2. 根据classid查找系统模型信息、根据系统模型获取数据表以及其分表信息。
3. 若classid为空或无法获取到对应系统模型、数据表时，则返回信息“来源错误”
4. 设置操作类型变量，若id为空则设置为“添加”。并生成uuid设为新id
5. 若id非空，则根据表名查找对应id数据。找到则设置操作类型为“修改”，找不到则设置为添加。
6. 根据操作类型，从系统模型数据中获取对应的可添加/修改字段。
7. 遍历获取到的字段集合，分别查询字段信息，再按照其所属的主表/分表以及infojson中的值、组成相应的insert/update语句，如果是修改，则将原数据复制到微客表，如果taskid不为空则设置威客id为taskid,否则生成UUID作为威客id，修改人，修改时间(weike\_id、modifyauthor、modifytime)，然后再更新原数据表的数据。
8. 在主表语句中，追加系统字段（推荐，头条，置顶，审核，发布时间）
9. 执行生成好的sql语句
10. 返回信息（注意把信息id最后放入flag中）

返回值格式：

{

"result": { //结果

"msg": "修改成功", //信息

"flag": "1" //直接存入信息id

}

}

* + 1. 其他转信息方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| translateToInfo | | String | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| Id | String | | 需要转换的id |
| title | String | | 需要转换的名字 |
| tbname | String | | 转换目标的表名 |
| fid | String | | 要转换的信息所属栏目id |

方法逻辑说明：

1. 检查参数值，任意值为空则返回“来源错误”
2. 根据fid查找栏目所绑定的系统模型id，通过系统模型id找到对应的信息表名作为 tbname的值。
3. 根据来源数据设置字段id、title，同时设置来源表名字段，执行sql语句添加转换记录。
4. 返回“操作成功”

pS :接口名称：机构转换信息接口url OrgTransferToinfo.do；

人员转换信息url：peopTransferToinfo.do

数据词典url：dataDicTransferToinfo.do

角色：RoleTransferToinfo.do

标签：TagTransferToinfo.do

* + 1. 信息转其他方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| infoTranslateToOthers | | int | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| Id | String | | 信息id |
| title | String | | 信息的名字 |
| tbname | String | | 转换目标的表名 |
| btype | String | | 转换的类型（人员、角色、栏目、标签、数据词典、机构） |
| origintable | String | | 来源表名字段 |
| Targettable | String | | 目标表名字段（多个表名以“，”隔开） |

方法逻辑说明：

1. 检查参数值id,title，btype，任意值为空返回”来源错误”
2. 根据id获取来源数据，若找不到数据则同样返回“来源错误”
3. 根据来源数据设置字段 主键id、名称，拼接SQL语句。
4. 根据id的值和通过classid获取的表名获取targettable的值，然后更新这个值（把原有的值附加新的表名用逗号隔开）。
5. 执行添加sql语句，向转换目标表增加记录。
6. 返回“操作成功”
   * 1. **查重接口**

**a.接口url:**

../setStaffTag.do

**b.接口说明:**

该接口所有表的查重用

**c.接口定义：**

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| checkRepeat | | Result | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| tbname | String | | 需要查重的信息表名 |
| title | String | | 查重的字段名 |
| value | String | | 字段值 |

**d.返回值示例与说明**

{

"result": { //结果

"msg": "名称重复", //成功转移数量

"flag": "1" //标识，1为重复，0为不重复

}

}

**e.功能实现说明**

① 查询数据表名为tbname的数据表，查询该表的字段名为参数title的字段，查找该字段值等于参数value的字段的数据，

1）若查找到的结果不为空则返回“名称重复”;

2)若查找到的结果为空则返回“名称可以使用”;

* + 1. 查询列表接口（根据列表项动态） +

**a.接口url:**

pub/admin/information/Infohtml/findInfoTempList.do

**b.接口说明:**

该接口用于查询对应栏目下的信息数据列表（动态）。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findInfoTempList | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| userid | String | 用户id |
| page | String | 当前页码 |
| num | String | 每页显示信息条数 |
| orderType | String | 排序类型 |
| tablechange | String | gd表示查询归档表 |
| searchAttr | String | 搜索方法的属性，如：  1为置顶、2为推荐、3为头条、4为未审核、5为已审核、6为投稿、7为签发、8为我的信息 |
| searchValue | String[] | 搜索方法的关键字数组 |
| searchField | String[] | 搜索方法的字段数组，即该表包含的字段，如：  标题、机构、副标题等。 |
| searchTime | String | 搜索方法的时间，如：  1天、2天、一周、1个月、3个月、6个月、1年。 |

**d.返回值示例与说明**

{

"page": { //分页信息

"page": 2, //当前页

"pages": 10, //总页数

"lines": 100, //信息总条数

"pageSize": "10", //每页显示信息数

"pageStart": 1 //起始页

},

"total": 1, //当前页信息数

"infos": [ //信息

{

……

},

{

……

}

]

}

**e.功能实现说明**

1. 检查参数classid、为空则返回信息“来源错误”。非空则通过classid找到栏目对象，通过栏目对象找到mid，通过mid找到系统模型，通过系统模型找到tbname和列表项。
2. 根据tbname找到相关分表
3. 如果tablechange非空，则根据tablechange修改各表名至对应归档表
4. 一次检测searchAttr、searchValue、searchField、searchTime参数，非空值、则根据对应数值拼接sql查询条件
5. 若orderType非空则拼接order by排序内容
6. 根据num和page拼接limit内容。若为空值则使用默认值
7. 执行生成好的sql语句、查询并返回数据
   * 1. 查询信息数据接口（根据内容项动态） +

**a.接口url:**

pub/admin/information/Infohtml/findTempInfo.do

**b.接口说明:**

该接口用于查询对应栏目下的信息数据（根据内容项动态）。

**c. 接口定义：**

|  |  |  |
| --- | --- | --- |
| 接口定义 | | |
| 接口名 | 对象名 | 返回值类型 |
| findTempInfo | Info | Json |
| 参数说明： | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目id |
| id | String | 信息id |
| method | String | method为”modify”时为修改信息查找，method为“show”时为显示信息时查找信息 |

**d.返回值示例与说明**

//信息

{

“infos”：{……}

}

**e.功能实现说明**

1. 接收参数id、method、classid，有任意一个空值则返回“来源错误”
2. 根据classid找到栏目信息中的系统模型mid
3. 判断method为“modify”还是为“show”

1.当method为“modify”时

1. 根据通过mid找到系统模型信息中的表名（tbname）以及录入项（enter）
2. 分解enter重新组合成字段拼接字符串：字段1,字段2,字段3
3. 根据id和tbname查询对应表中的enter所示字段以及系统字段，返回 数据

2.当method为“show”时

1）根据通过mid找到系统模型信息中的表名（tbname）以及内容项（tempvar）

2）分解tempvar重新组合成字段拼接字符串：字段1,字段2,字段3

3）根据id和tbname查询对应表中的tempvar所示字段，返回数据

* 1. 系统基本设置
     1. 获取系统参数方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| getZjxxSysParam | | infor | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| 参数名称 | 参数类型 | | 备注 |
| title | String | | 参数英文名、扩展变量名 |

方法逻辑说明：

①检测参数title，当值不为空时加入查询条件，直接使用精确查找。

②先在缓存中查询对应数据，如果存在则返回该数据。

③如果缓存中找不到，则将条件传入mybatis查询接口，组成查询语句。

④执行得到查询结果对象，如果其iscache属性为1，则将其写入缓存，返回该数据。

* + 1. 修改系统参数方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法定义 | | | |
| 方法名 | | 返回值类型 | |
| setZjxxSysParam | | Result | |
| 参数说明： | | | |
| 参数名称 | 参数类型 | | 备注 |
| title | String | | 参数英文名 |
| content | String | | 系统参数设置的内容 |
| iscache | String | | 是否写入缓存 |

方法逻辑说明：

①检测参数title、若为空值则返回“title为空”的错误信息

②若title不为空、则根据title查询对应系统参数/扩展变量数据。若无法找到对应数据，则返回“未找到对应参数”的错误信息

③若成功找到对应数据，则根据其他参数更新数据库中的数据

④服务器上根据key=title查找application的缓存。找到则清楚该缓存。找不到则不管。

⑤返回更新成功信息

1. 对象
   1. 系统用到的对象
      1. 系统参数类对象

系统参数类用到的对象参数表：类ZjxxSysparam 对象一般用zjxxSysparam。

|  |  |  |
| --- | --- | --- |
| 系统参数类用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 系统参数id |
| title | String | 参数英文名 |
| cname | String | 参数中文名 |
| content | String | 内容 |
| type | String | 参数类型 |
| iscache | Boolean | 是否写入缓存，1写入，0不写入 |

* + 1. 模型数据表类对象

模型数据表用到的对象参数表：类ZjxxEnewstable 对象一般用zjxxEnewstable。

|  |  |  |
| --- | --- | --- |
| 模型数据表类用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| tid | String | 数据表ID |
| tbname | String | 数据表名 |
| tname | String | 数据表标识 |
| tbtype | int | 表类型 |
| tbs | String | 对应字段 |
| tsay | String | 备注 |
| ismaster | Boolean | 是否是主表 |
| masterb | String | 主表名称 |
| parenttb | String | 父表名称 |
| sontb | String | 子表集合 |
| datatbs | String | 分表集合 |
| torder | int | 顺序 |
| yhid | String | 使用优化方案ID |
| orgid | String | 机构ID |

* + 1. 字段类对象

字段类用到的对象参数表：类ZjxxEnewsf对象一般用zjxxEnewsf。

|  |  |  |
| --- | --- | --- |
| 字段类用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| fid | String | 字段ID |
| f | String | 字段名 |
| fname | String | 字段标识 |
| fform | String | 输入表单显示元素 |
| fhtml | String | 输入表单替换html代码 |
| fzs | String | 注释 |
| isadd | Boolean | 是否系统字段 |
| isshow | Boolean | 是否显示字段 |
| iscj | Boolean | 是否为采集项 |
| cjhtml | String | 采集表单替换html |
| myorder | Short | 显示顺序 |
| ftype | String | 字段类型 |
| flen | String | 字段长度 |
| dotemp | Boolean | 是否为模板项 |
| tid | String | 数据表ID |
| tbname | String | 数据表名 |
| savetxt | Boolean | 是否内容存文本 |
| fvalue | String | 字段初始值 |
| iskey | Boolean | 是否为索引字段 |
| tobr | Boolean | 内容回车替换成换行符 |
| dohtml | Boolean | 内容支持html代码 |
| qfhtml | String | 投稿表单替换html代码 |
| isonly | Boolean | 字段值是否唯一 |
| linkfieldval | String | 关联显示值字段名 |
| samedata | Boolean | 关联是否数据同步 |
| fformsize | String | 表单元素长度 |
| tbdataf | Boolean | 是否为分表字段 |
| ispage | Boolean | 是否分页字段 |
| linkfieldtb | String | 关联数据表明 |
| linkfieldshow | String | 关联显示字段 |
| editorys | Boolean | 编辑器样式 |
| issmalltext | Boolean | 是否简介字段 |
| jsonstring | String | 选项 |
| organization | String | 所属机构ID |

* + 1. 系统模型类对象

系统模型类用到的对象参数表：类ZjxxModel对象一般用zjxxModel。

|  |  |  |
| --- | --- | --- |
| 系统模型类用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| mid | String | 模型ID |
| mname | String | 模型名称 |
| mtemp | String | 录入表单模板 |
| mzs | String | 注释 |
| cj | String | 采集项 |
| enter | String | 录入项 |
| tempvar | String | 内容木板箱 |
| sonclass | String | 终极栏目ID集合 |
| searchvar | String | 搜索项 |
| tid | String | 数据表ID |
| tbname | String | 数据表名 |
| qenter | String | 投稿项 |
| mustqenterf | String | 必填项 |
| qmtemp | String | 前台投稿表单模板 |
| listandf | String | 结合项 |
| setandf | Boolean | 结合相匹配 |
| listtempvar | String | 列表模板项 |
| qmname | String | 模型别名 |
| canaddf | String | 可增加项 |
| caneditf | String | 可修改项 |
| definfovoteid | Short | 信息预设投票ID |
| showmod | Boolean | 是否显示到前台导航 |
| usemod | Boolean | 是否启用 |
| myorder | Short | 排序 |
| orderf | String | 排序项 |
| isdefault | Boolean | 是否默认模型 |
| listfile | String | 信息列表名称 |
| printtempid | Short | 打印模板ID |
| enterjson | String | 录入项类型 |
| orgid | String | 所属机构 |
| uk | String | 联合主键 |
| clunmid | String | 栏目ID |

* + 1. 基础字段类对象

基础字段类用到的对象参数表：类ZjxxBaseTableField对象一般用zjxxBaseTableField。

|  |  |  |
| --- | --- | --- |
| 基础字段类用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| columnName | String | 字段名 |
| dataType | String | 字段类型 |
| characterMaxImumLength | String | 字段长度 |
| columnComment | String | 字段备注 |
| isNullable | String | 是否为空 |
| columnKey | String | 是否为主键 |

* 1. 栏目用到的对象
     1. 栏目主表对象

栏目用到的对象参数表：ZjxxEnewscolumn

|  |  |  |
| --- | --- | --- |
| 栏目类用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目ID |
| classname | String | 栏目名称 |
| bname | String | 栏目别名 |
| columntype | Integer | 栏目类型 |
| islast | Integer | 是否终极栏目 |
| isgd | Integer | 是否归档 |
| isinfo | Integer | 是否自动转换为信息 |
| showclass | Integer | 是否显示到导航 |
| openadd | Integer | 是否开放投稿 |
| myorder | Integer | 排序 |
| islist | Integer | 页面显示模式 |
| columnclass | Integer | 栏目页模式 |
| showdt | Integer | 内容页模式 |
| isflash | Integer | 内容条数增删时，列表页是否自动刷新 |
| maxnum | Integer | 显示总记录数 |
| lencord | Integer | 每页显示信息数 |
| linknum | Integer | 相关链接数 |
| classtype | String | 栏目文件扩展名 |
| filetype | String | 内容页文件扩展名 |
| is\_zt | Integer |  |
| author | String | 作者 |
| top | Integer | 是否置顶 |
| onclick | Integer | 点击数 |
| downnum | Integer | 每行下载地址数 |
| onlinenum | Integer | 每行在线观看地址数 |
| addinfofen | Integer | 前台投稿加点数 |
| listdt | Integer | 列表模式 |
| doctime | Integer | 信息归档时间 |
| stpnum | Integer | 生成静态页数 |
| addreinfo | Integer | 发布信息生成内容页 |
| modid | String | 所属系统模型ID |
| isshow | Integer | 是否显示 |
| openpl | Integer | 是否开启评论 |
| plmodids | String | 评论系统模型id集合 |
| orgid | String | 所属机构id |
| trainfo | Integer | 转换为信息 |
| columnid | String | 自动转换信息栏目 |
| pid | String | 父栏目id |

* + 1. 栏目副表对象

栏目用到的对象参数表：ZjxxEnewsColumnPart1

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目ID |
| sametitle | Integer | 是否检测标题重复 |
| repreinfo | Integer | 是否生成上一篇信息 |
| haddlist | Integer | 发布生成列表 |
| listorder | String | 管理信息排序方式 |
| reorder | String | 列表式页面排序方式 |
| nreclass | Integer | 是否生成栏目页 |
| nreinfo | Integer | 是否生成内容页 |
| nrejs | Integer | 是否生成JS调用 |
| nottobq | Integer | 标签是否调用 |
| wapapplychild | Integer | WAP是否应用于子栏目 |
| filenameqz | String | 内容文件名前缀 |
| newspath | String | 内容页日期目录形式 |
| filename | Integer | 内容页文件命名形式 |
| newline | Integer | 最新信息JS显示数 |
| hotline | Integer | 热门信息JS显示数 |
| goodline | Integer | 推荐信息JS显示数 |
| hotplline | Integer | 热门评论信息JS显示数 |
| firstline | Integer | 头条信息JS显示数 |
| definfovoteid | String | 信息预设投票ID |
| tid | String | 数据表ID |
| classtempid | String | 封面模板ID |
| dtlisttempid | String | 动态列表模板ID |
| jtlisttempid | String | 静态列表模版ID |
| wid | String | 所属Word模板 |
| searchtempid | String | 搜索模板ID |
| wapstyleid | String | WAP模板ID |
| jstempid | String | JS模板ID |
| listtempid | String | 列表模板ID |
| lclassid | String | 终极栏目ID |
| pltempid | String | 评论模板ID |
| newstempid | String | 内容模板ID |
| did | String | 动态选择栏目ID |
| gdid | String | 归档方案ID |

* + 1. 栏目副表 权限

栏目用到的对象参数表：ZjxxEnewsColumnPart2

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目ID |
| qaddgroupid | String | 可投稿用户 |
| q\_whitelist | String | 白名单（投稿） |
| q\_blacklist | String | 黑名单（投稿） |
| seegroupid | String | 可查看用户 |
| s\_whitelist | String | 白名单（查看） |
| s\_blacklist | String | 黑名单（查看） |
| modgroupid | String | 可修改用户 |
| m\_whitelist | String | 白名单（修改） |
| m\_blacklist | String | 黑名单（修改） |
| cgtoinfo | Integer | 栏目访问权限应用于信息 |
| checkqadd | Integer | 前台投稿是否要审核 |
| qaddlist | Integer | 投稿生成列表 |
| qaddshowkey | Integer | 投稿是否开启验证码 |
| adminqinfo | Integer | 管理投稿 |
| qeditchecked | Integer | 修改投稿是否需要审核 |
| checkpl | Integer | 评论是否需要审核 |
| checked | Integer | 信息默认是否审核 |

* + 1. 栏目副表 大字段

栏目用到的对象参数表：ZjxxEnewsColumnPart3

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| classid | String | 栏目ID |
| pagecon | String | 页面内容 |
| classtext | String | 栏目页面内容 |
| ttids | String | 栏目选择标题分类ID |
| wburl | String | 外部栏目链接地址 |
| ipath | String | 自定义内容页存放目录 |
| listselect | String | 列表条件选择 |
| subcondition | String | 专题条件 |
| classimg | String | 栏目缩略图 |
| classpagekey | String | 页面关键字 |
| intro | String | 栏目简介 |
| classpath | String | 栏目存放目录 |
| sonclass | String | 终极栏目ID集合 |
| lfiel | String | 列表接口下发字段 |
| sonids | String | 子栏目ID集合 |
| classurl | String | 栏目绑定域名 |
| yiscolumn | Integer | 域名应用于子栏目 |
| tbname | String | 数据表名 |

* 1. 模版类用到的对象
     1. 列表模版类对象

列表类模版用到的对象参数表：类ZjxxListTemp 对象一般用listTemp。

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| tempid | String | 模版ID |
| tempname | String | 模版名称 |
| temptext | mediumtext | 模版内容 |
| subnews | int | 截取自字段数 |
| isdefault | tinyint(1) | 是否默认 |
| listvar | text | 列表内容模板 |
| rownum | int | 每次显示 |
| modid | String | 所属系统模型ID |
| showdate | String | 时间显示格式 |
| subtitle | int | 标题截取字数 |
| orgid | String | 机构ID |
| docode | tinyint(1) | 是否使用程序代码 |

* + 1. 内容模版类对象

封面类模版用到的对象参数表：类ZjxxContentTemp 对象一般用contentTemp。

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| tempid | String | 模版ID |
| tempname | String | 模版名称 |
| temptext | mediumtext | 模版内容 |
| isdefault | int | 是否默认模板 |
| showdate | String | 时间显示格式 |
| modid | String | 所属系统模型ID |
| orgid | String | 机构ID |

* + 1. 封面模版类对象

封面类模版用到的对象参数表：类ZjxxCoverTemp 对象一般用coverTemp。

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| tempid | String | 模版ID |
| tempname | String | 模版名称 |
| temptext | mediumtext | 模版内容 |
| orgid | String | 机构ID |

* 1. 用户模块用到的对象
     1. 机构类对象

机构类用到的对象参数表：类ZjxxOrgnization 对象一般用orgnization。

|  |  |  |
| --- | --- | --- |
| 机构类用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 主键ID |
| title | String | 名称 |
| description | String | 描述 |
| did | String | 数据词典ID（机构类别） |
| rids | String | 角色ids |
| parentOrg | String | 父级机构 |
| Legal0rg | String | 机构法人 |
| orgSize | String | 规模 |
| childOrg | String | 子级机构 |
| createDate | Date | 创建日期 |
| orgAddress | String | 机构地址 |
| orgEmail | String | 机构邮箱 |
| orgPhone | String | 机构电话 |
| isdelete | int | 1删除 0未删除 |
| menuid | String | 菜单id |
| linkruleid | String | 转化规则ID |

* + 1. 人员类对象

人员类用到的对象参数表：类ZjxxPeople 对象一般用people。

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 人员ID |
| title | String | 姓名 |
| orgid | String | 所属机构 |
| account | String | 用户名 |
| password | String | 密码 |
| idNumber | String | 身份证号 |
| rorder | String | 顺序 |
| rTime | String | 注册时间 |
| email | String | 邮箱 |
| bids | String | 标签ID |
| t\_time | Date | 产生tocken时间 |
| tocken | String | 登录标志 |
| rids | String | 角色ids(角色信息) 多个用|隔开 |
| linkruleid | String | 转化规则ID |

* + 1. 角色类对象

角色类用到的对象参数表：类ZjxxRole 对象一般用role。

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 角色ID |
| title | String | 角色名称 |
| description | String | 角色描述 |
| pid | String | 父级角色id |
| rorder | int | 角色顺序 |
| isdelete | int | 是否删除 |
| cid | String | 子角色id |
| linkruleid | String | 转化规则ID |
| orgid | string | 机构ID |

* + 1. 菜单类对象

菜单类用到的对象参数表：类ZjxxMenu 对象一般用menu。

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 菜单ID |
| title | String | 菜单名称 |
| description | String | 菜单描述 |
| isFinal | int | 是否终极菜单 |
| parentMenu | String | 所属父级菜单 |
| childMenu | String | 子级菜单 |
| pageUrl | String | 页面url |
| menuorder | int | 菜单顺序 |
| seegroupid | text | 可查看用户组 |
| whitelist | text | 白名单 |
| blacklist | text | 黑名单 |
| linkruleid | String | 转化规则ID |
| orgid | String | 机构id |

* + 1. 数据词典类对象

数据词典类用到的对象参数表：类ZjxxDictionary 对象一般用dictionary 。

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 数据词典ID |
| title | String | 数据词典名称 |
| description | String | 数据词典描述 |
| orderNum | int | 数据词典顺序 |
| pid | String | 父级词典id |
| cid | String | 子级词典id |
| linkruleid | String | 转化规则ID |

* + 1. 标签类对象

标签类用到的对象参数表：类ZjxxTag 对象一般用tag。

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| id | String | 标签id |
| title | String | 标签名称 |
| description | String | 标签描述 |
| torder | int | 顺序 |
| linkruleid | String | 转化规则ID |

* 1. 全局对象
     1. 分页实体类对象

分页实体类对象参数表：类Page,对象一般用page

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| pageStart | int | 起始页 |
| pageSize | int | 每页显示的数目 |
| lines | int | 总条数 |
| pages | int | 总页数 |
| page | int | 当前页码 |

* + 1. 返回信息类对象

分页实体类对象参数表：类Page,对象一般用page

|  |  |  |
| --- | --- | --- |
| 列表类模版用到的对象参数表 | | |
| 参数名称 | 参数类型 | 备注 |
| total | int | 当前页查找的实际信息数 |
| page | String | 分页对象信息存放字段 |
| infos | String | 信息，查找到的满足条件的信息集合整合到这个字段属性中 |

1. 附录
   1. 信息栏目转化字段对照表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **信息转换字段对应** | | | | | |
| **表名** | **主键** | **名称** | **父项** | **联动字段** | **需处理字段** |
| 1.基础信息表 | id | title | classid | linktables | 无 |
| 2.栏目表 | classid | classname | bclassid | origintable | sonids |
| 3.机构表 | id | title | parent\_org | origintable | child\_org |
| 4.数据词典表 | id | title | pic | origintable | cid |
| 5.角色表 | id | title | pid | origintable | cid |
| 6.人员表 | id | title | 无 | origintable | 无 |
| 7.菜单表 | id | title | parent\_menu | origintable | child\_menu |
| 8.标签表 | id | title | 无 | origintable | 无 |
|  |  |  |  |  |  |