开发工具 idea

[\\lichun\soft\java\idea](\\\\lichun\\soft\\java\\idea)

需要配置maven,tomcat

引入的java包，统一在pom.xml中管理

例:

<dependency>  
 <groupId>mysql</groupId>  
 <artifactId>mysql-connector-java</artifactId>  
 <version>5.1.25</version>  
</dependency>

在pom.xml plugin标签中注册新增的模块以及sql.xml文件

<configuration>  
 <sources>  
 <source>${basedir}/SystemManagement/</source>  
 <source>${basedir}/workflowEngine/</source>  
 <source>${basedir}/SatelliteReport/</source>  
 <source>${basedir}/SatelliteGraph/</source>  
 <source>${basedir}/common/</source>  
</configuration>

<resource> <directory>${basedir}/SatelliteGraph/sql/mapper</directory>  
 <targetPath>sql/mapper</targetPath>  
</resource>

**配置文件在phalaenopsis/common/conf 下面**

1 applicationContext.xml 配置上下文

其中

<bean id="exceptionHandler"class="phalaenopsis.common.method.MyExceptionHandler" />

配置统一的异常拦截器

程序中不要自己捕获异常，不然事务无法回滚

2 applicationContext\_dao.xml 配置mybatis,实体类映射

3 applicationContext\_datasource.xml 配置数据库连接及事务

4 config.properties 配置固定参数

例

#上传文件存放路径  
UploadFolder=/UploadFiles/

phalaenopsis.common.entity AppSettings 中获取

5 dispatcher-servlet.xml 配置消息转换器 以及拦截器

<mvc:interceptors>  
 <mvc:interceptor>  
 <mvc:mapping path="/\*\*/\*" />  
 <bean class="phalaenopsis.common.method.CrossInterceptor">  
 <property name="ignoreUrls"  
 value="/api;/Sys/Auth/Login;/Sys/Auth/loginForLC;/Sys/Auth/Test;/Foundation/DataDictionary/DownloadApp;/Foundation/DataDictionary/GetLatestVersion;/SatelliteGraph/Mobile/CheckConnection;/LawCase/LawCaseService/GetSignLink;/SatelliteGraph/Service/test"></property>  
 </bean>  
 </mvc:interceptor>  
 <!-- liulang add start -->  
 <mvc:interceptor>  
 <mvc:mapping path="/\*\*" />  
 <bean class="phalaenopsis.common.interceptor.LogInterceptor">  
 </bean>  
 </mvc:interceptor>  
 <!-- liulang add end -->  
</mvc:interceptors>

所有前端发送的请求,会被拦截进入前置方法，如果身份票据验证不通过并且url不在ignoreUrls中，则无法访问服务

6 ehcache.xml 设置用户缓存，系统缓存

7 jdbc.properties 数据库连接

jdbc.kotei.url =   
jdbc.kotei.username =

jdbc.kotei.password =

8 workflow-config.xml 工作流配置

**基本结构**

**Controller层 公布前端访问的接口**

类名上加入标签

@Controller  
@RequestMapping("/LawCase/LawCaseService")

使用@Autowired 实例化service

@Autowired  
private LawCaseCoordinate coordinateService;

例

/\*\*  
 \* 获取案件坐标数据.  
 \* @param id 案件Id号  
 \* @return 返回坐标信息  
 \*/  
@RequestMapping(value = "/GetCaseShape", method = RequestMethod.GET)  
@ResponseBody  
public Polygon getCaseShape(@PathParam("id") final String id) {  
 return coordinateService.getCaseShape(id);  
}

/\*\*  
 \* 删除实测坐标信息.   
 \* @param info 案件信息  
 \* @return 返回是否成功  
 \*/  
@RequestMapping(value = "/DeleteRealCoordinate", method = RequestMethod.POST)  
@ResponseBody  
public boolean deleteRealCoordinate(@RequestBody final CaseBaseInfo info) {  
 return coordinateService.deleteCoordinate(info);  
}

**Serice层 实现业务逻辑**

类名上加入标签@Service("lawCaseService")

使用@Autowired 实例化接口

@Autowired  
private IWorkFlowDao workflowDao;

在方法上加入@Transactional 开启事务

**Dao层 接口**

类名加入标签@Repository("iLawCaseDao")

**Entity 实体**

**Sql/mapper xml 编写sql**

<mapper namespace="phalaenopsis.lawcase.dao.ILawCaseServiceDao">

Xml namespace指向接口

在common/method 下面有一些工具类

例ExportExcel 导出excel

common/method/Tools

DateUtils GuidHelper StrUtil

Common/temple/Generator 自动生成实体，增删改查