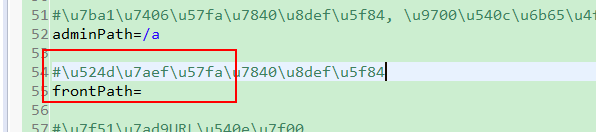
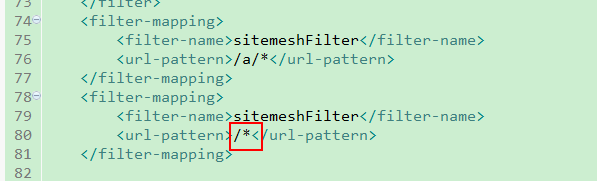
# 项目路径更改，前端改为根目录

jeesite.properties



web.xml



# 注解方式sql语句

public interface UserMapper {

/\*

\* 这是基于注解的映射方式，实现对数据的增删改查，将sql语句直接写在注解的括号中

\* 这是一个接口，其不需要类去实现它

\* 下边分别是插入，删除，修改，查询一个记录，查询所有的记录

\* \*/

@Insert("insert into users(name,age) values(#{name},#{age})")

public void insertT(User user);

@Delete("delete from users where id=#{id}")

public void deleteById(int id);

@Update("update users set name=#{name},age=#{age} where id=#{id}")

public void updateT(User user);

@Select("select \* from users where id=#{id}")

public User getUser(int id);

@Select("select \* from users")

public List<User> getAllUsers();

}

# 后台退出页面自定义

在com.thinkgem.jeesite.modules.sys.web. LoginController加入退出方法

/\*\*

\* 退出登录

\*/

@RequestMapping(value = "${adminPath}/logout", method = RequestMethod.***GET***)

**public** String logout(HttpServletRequest request, HttpServletResponse response) {

UserUtils.*getSubject*().logout();

**return** "redirect:" + adminPath + "/login";

}



# 自定义代码生成路径

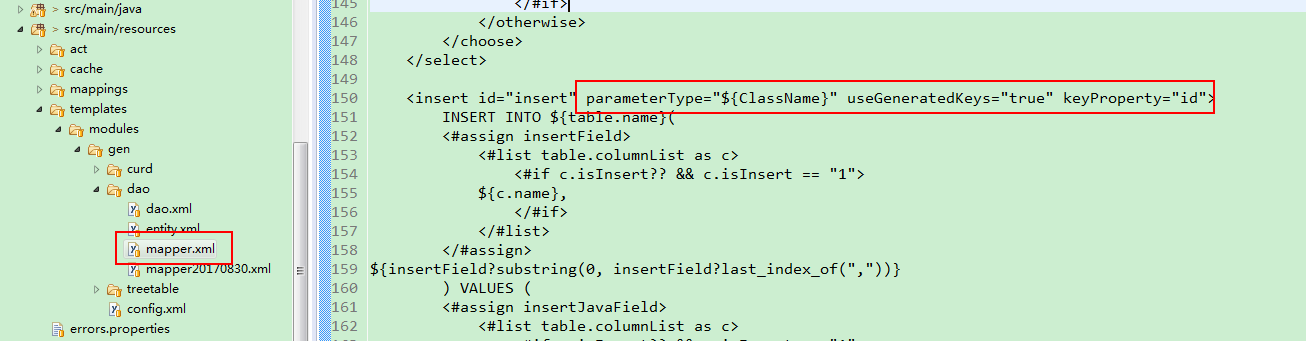




# 数据库时间以varchar存储的，前端改掉日期格式化

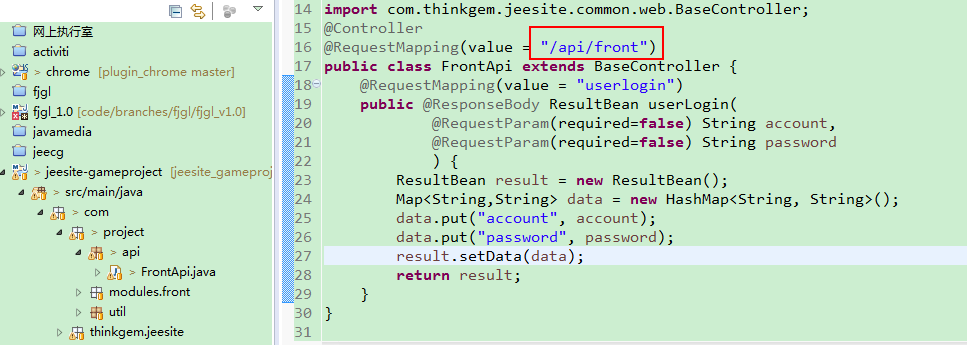
# 自定义id生成策略

id生成策略更改后，新加的数据id字段会为以前的uuid是不正确的，解决方法，修改mybatis加入数据后返回新加的id



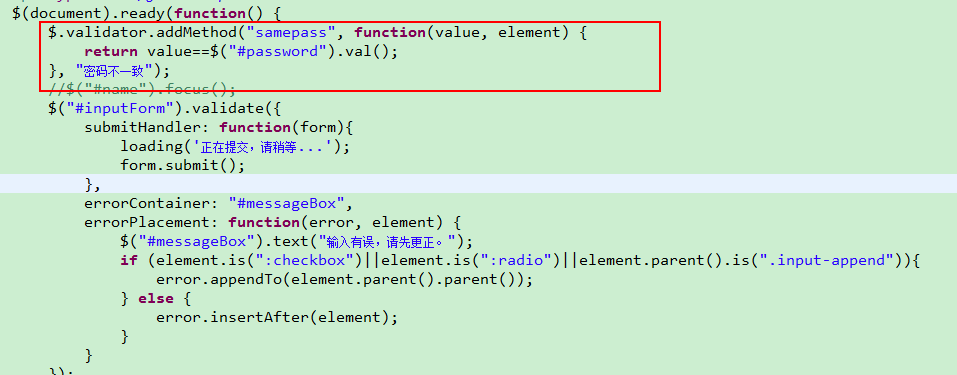
# 自定义API接口

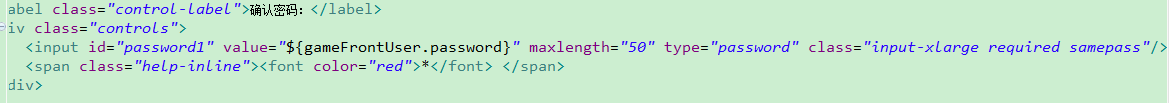
访问路径不要带有访问权限的路径



# 删除常量在BaseEntity里面

前端js自定义验证



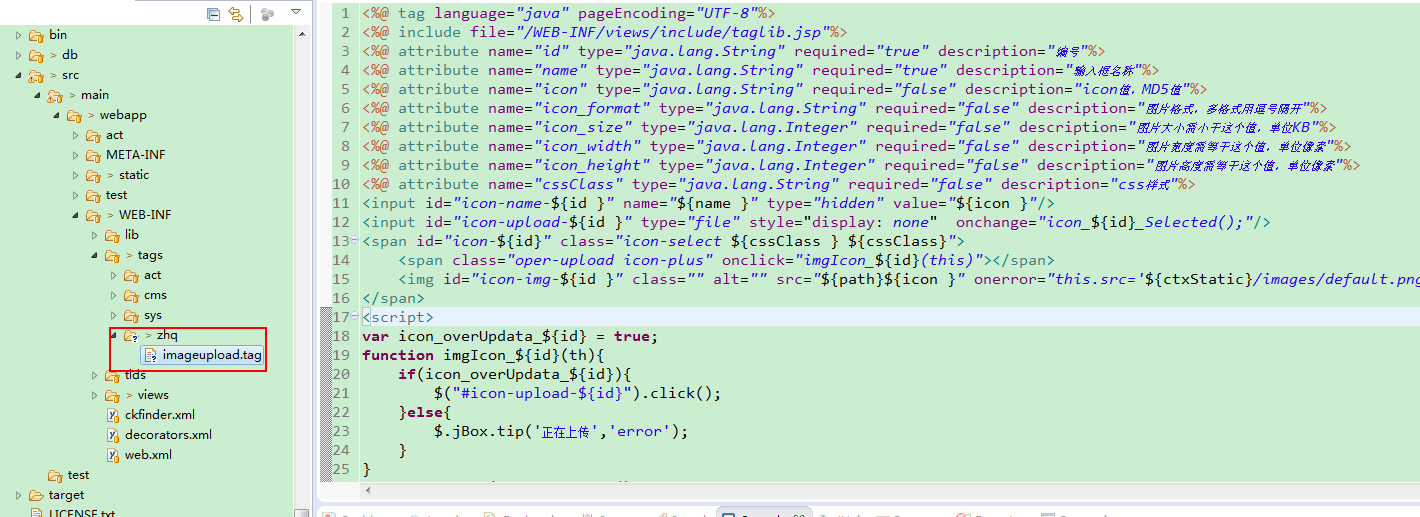


# 自定义数据类型，用于生产代码，同时自定义标签

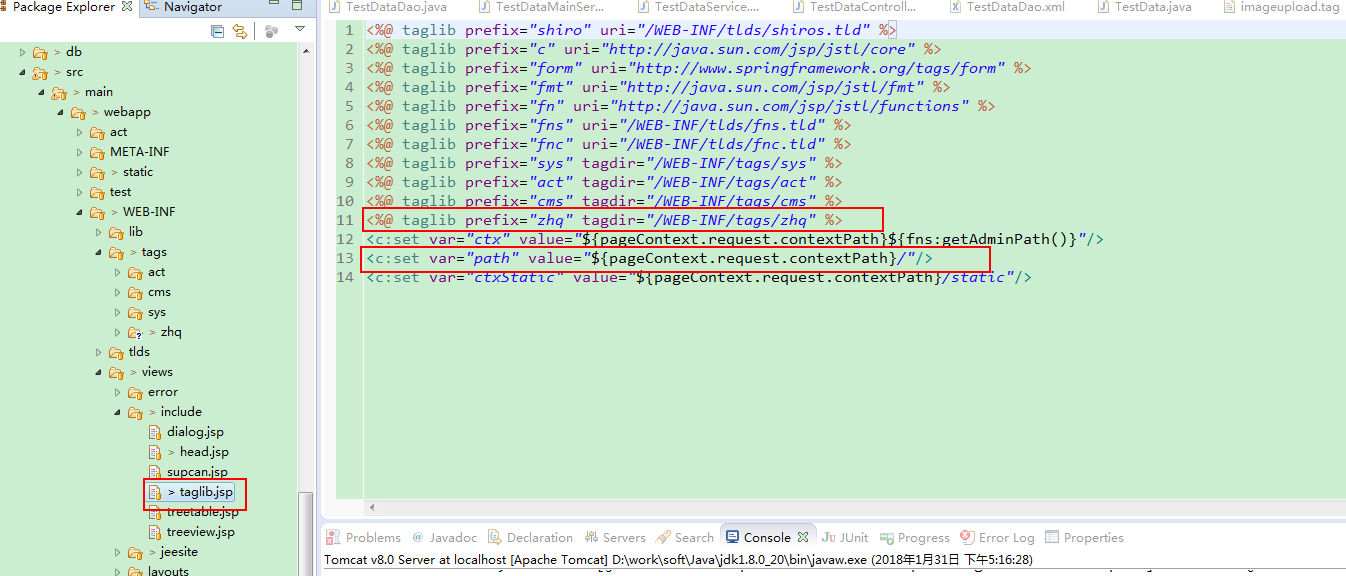
1. 加入自定义类型



1. 加入自定义标签

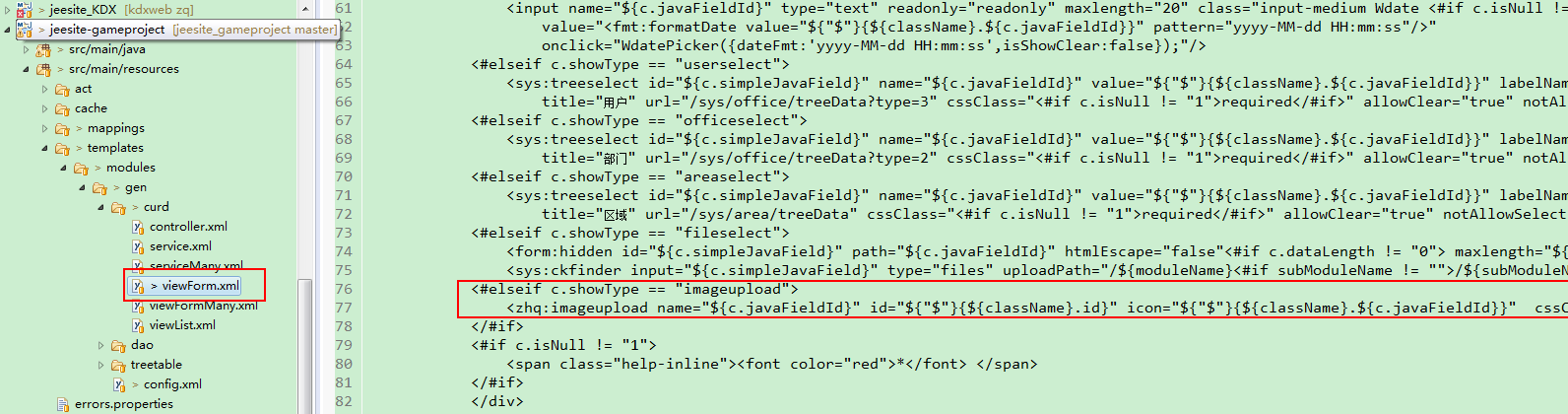


1. 引入标签

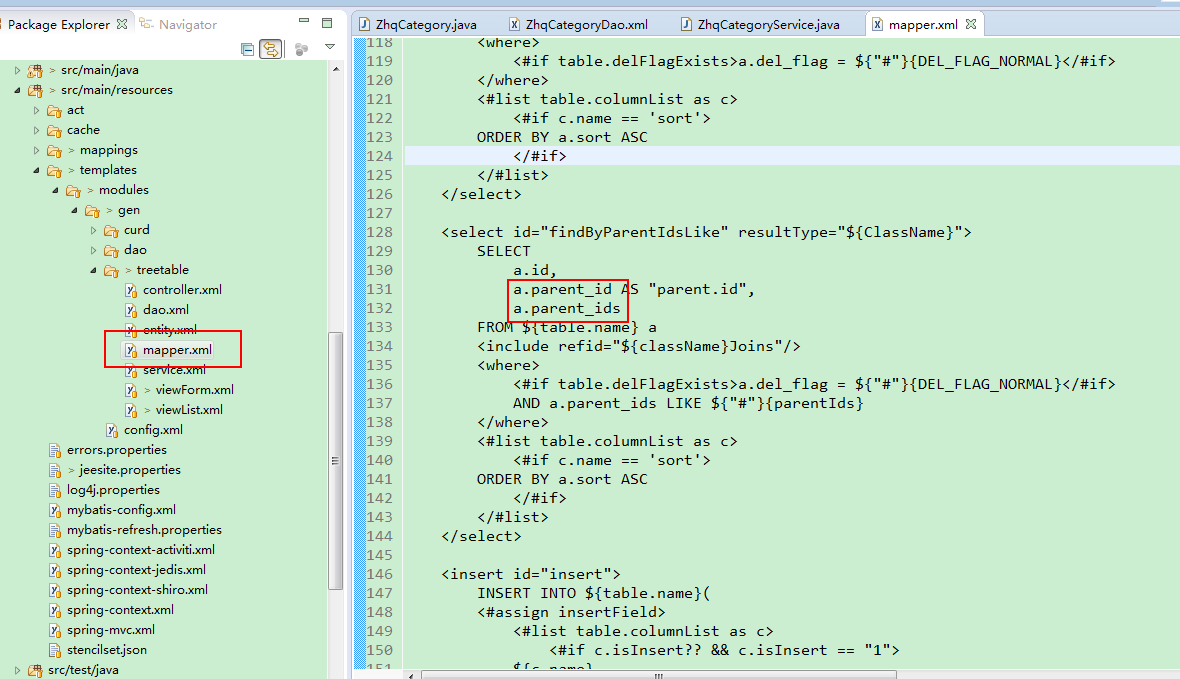


其中可以自定义常量

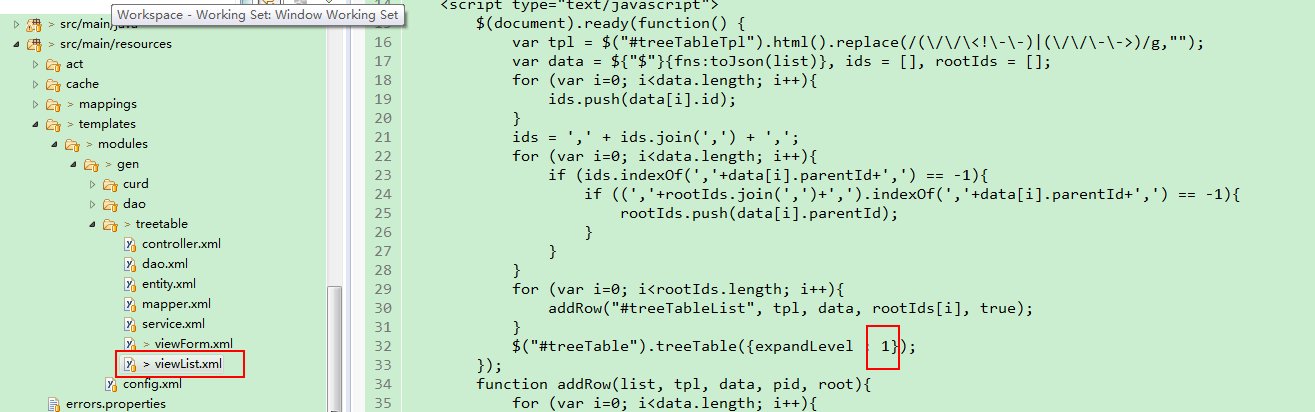
1. 更改页面模板



# 树形结构自定义pid和pids字段需手动更改，改为pid或path



# 树列表展开级别

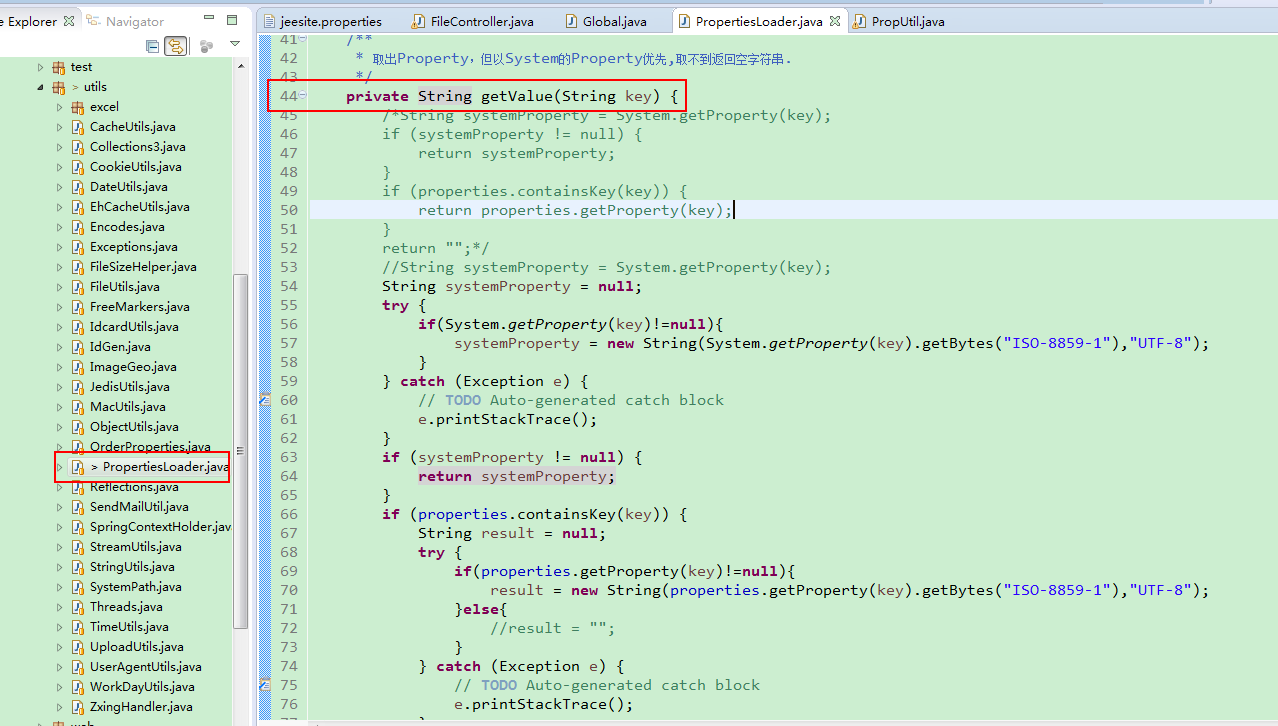


多级级联动菜单实现

# jeesite配置的读取

Global.getConfig("projectPath");

# 配置读取中文乱码的问题



/\*\*

\* 取出Property，但以System的Property优先,取不到返回空字符串.

\*/

**private** String getValue(String key) {

/\*String systemProperty = System.getProperty(key);

if (systemProperty != null) {

return systemProperty;

}

if (properties.containsKey(key)) {

return properties.getProperty(key);

}

return "";\*/

//String systemProperty = System.getProperty(key);

String systemProperty = **null**;

**try** {

**if**(System.*getProperty*(key)!=**null**){

systemProperty = **new** String(System.*getProperty*(key).getBytes("ISO-8859-1"),"UTF-8");

}

} **catch** (Exception e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**if** (systemProperty != **null**) {

**return** systemProperty;

}

**if** (properties.containsKey(key)) {

String result = **null**;

**try** {

**if**(properties.getProperty(key)!=**null**){

result = **new** String(properties.getProperty(key).getBytes("ISO-8859-1"),"UTF-8");

}**else**{

//result = "";

}

} **catch** (Exception e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

//return properties.getProperty(key);

**return** result;

}

**return** "";

}