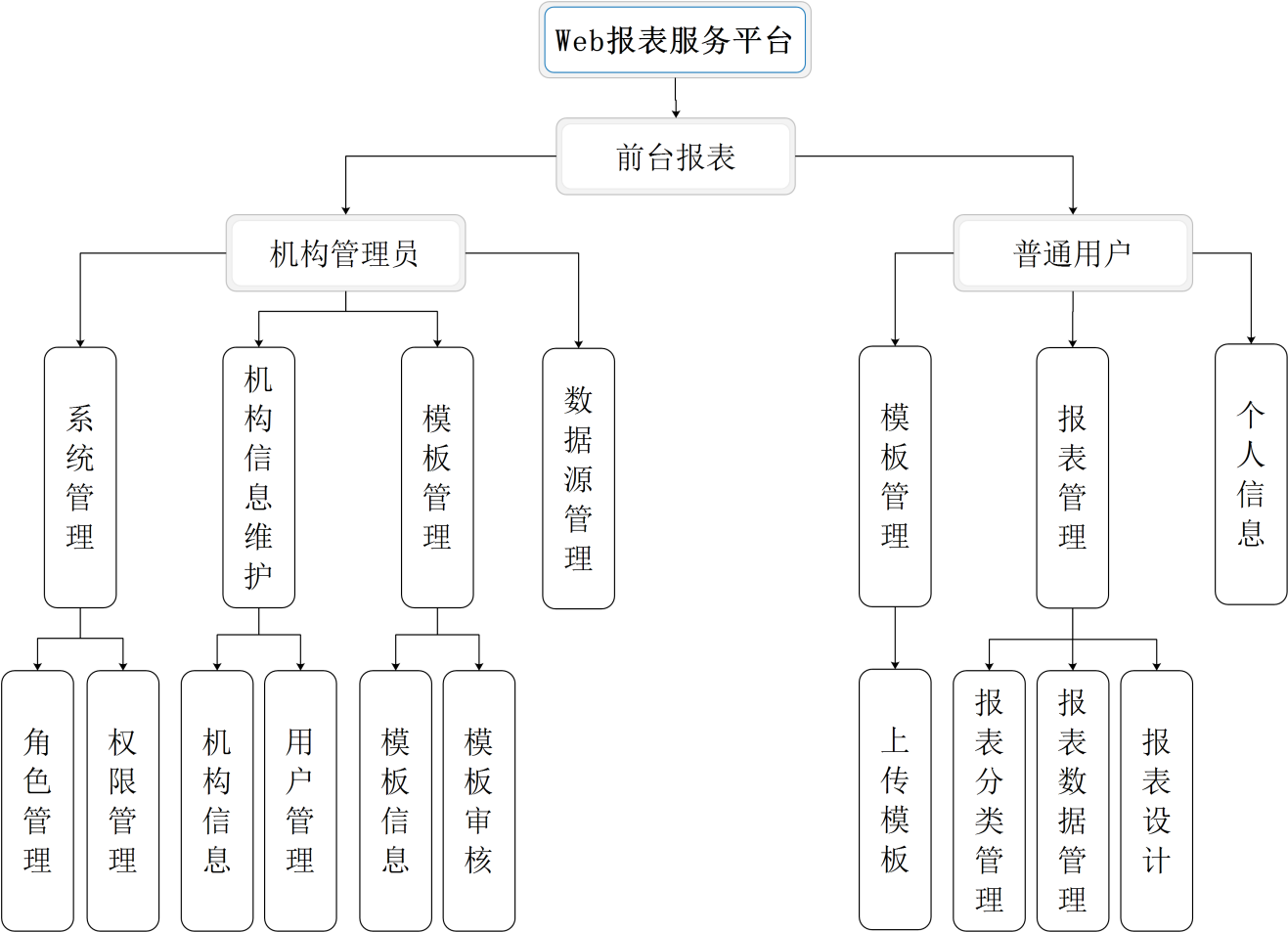
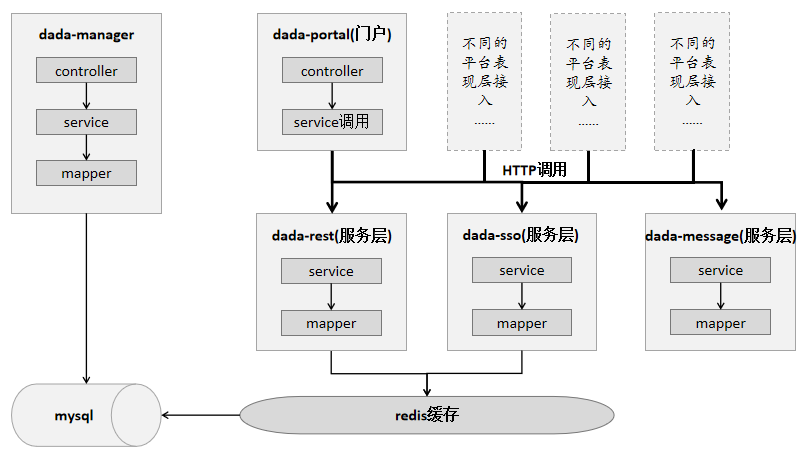
# 前端平台框架搭建

## 平台工程搭建（前端框架搭建）

### 1.系统架构分析





可以把业务逻辑提取出来，发布服务，供pc、移动端公用。

好处：

1. 提高代码复用度
2. 可以灵活的进行分布式部署。
3. 系统之间耦合度低
4. 团队之间开发不冲突。

缺点：

1. 需要发布Webservice，系统之间需要远程调用。
2. 工作量增加。
3. 需要系统之间协作才能完成整个业务流程，系统之间调用变的复杂。

需要创建两个工程

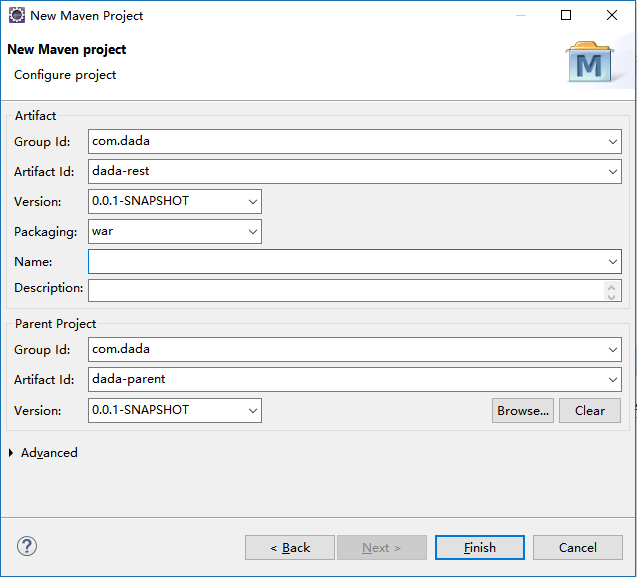
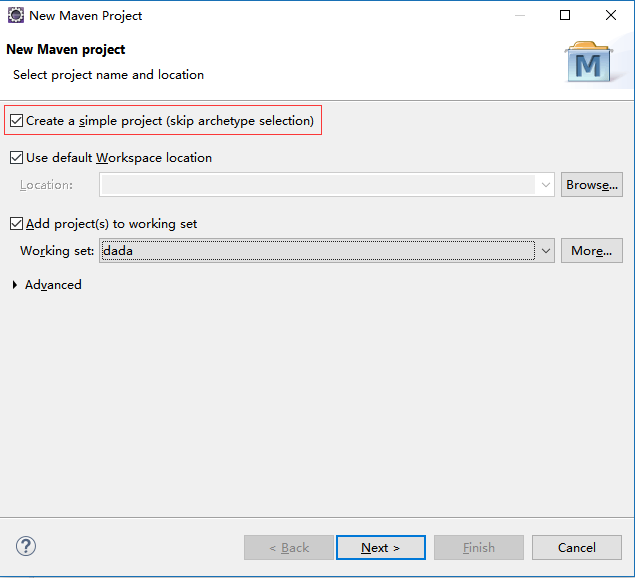
服务层：dada-rest：没有表现层，只有业务逻辑。需要发布服务。

表现层：dada-portal：只有表现层，没有业务逻辑，不需要连接数据库。

表现层和服务层通信，使用Webservice进行通信。Restful形式的Webservice。http+json数据。

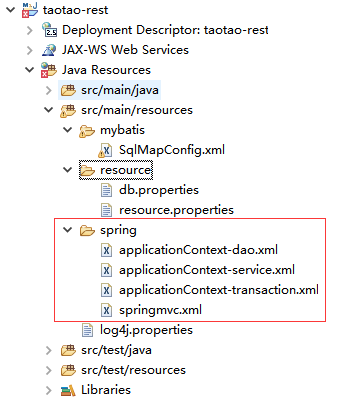
### 2.系统服务层搭建：dada-rest

#### 创建工程



#### 框架整合

整合ssm框架（可以参考taotao-manager工程）



修改说明：（针对每个文件进行修改说明）

applicationContext-service.xml、applicationContext-transaction.xml、springmvc.xml

Web.xml文件复制：修改两个内容--项目名称、拦截路径、控制器

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

#### Pom文件

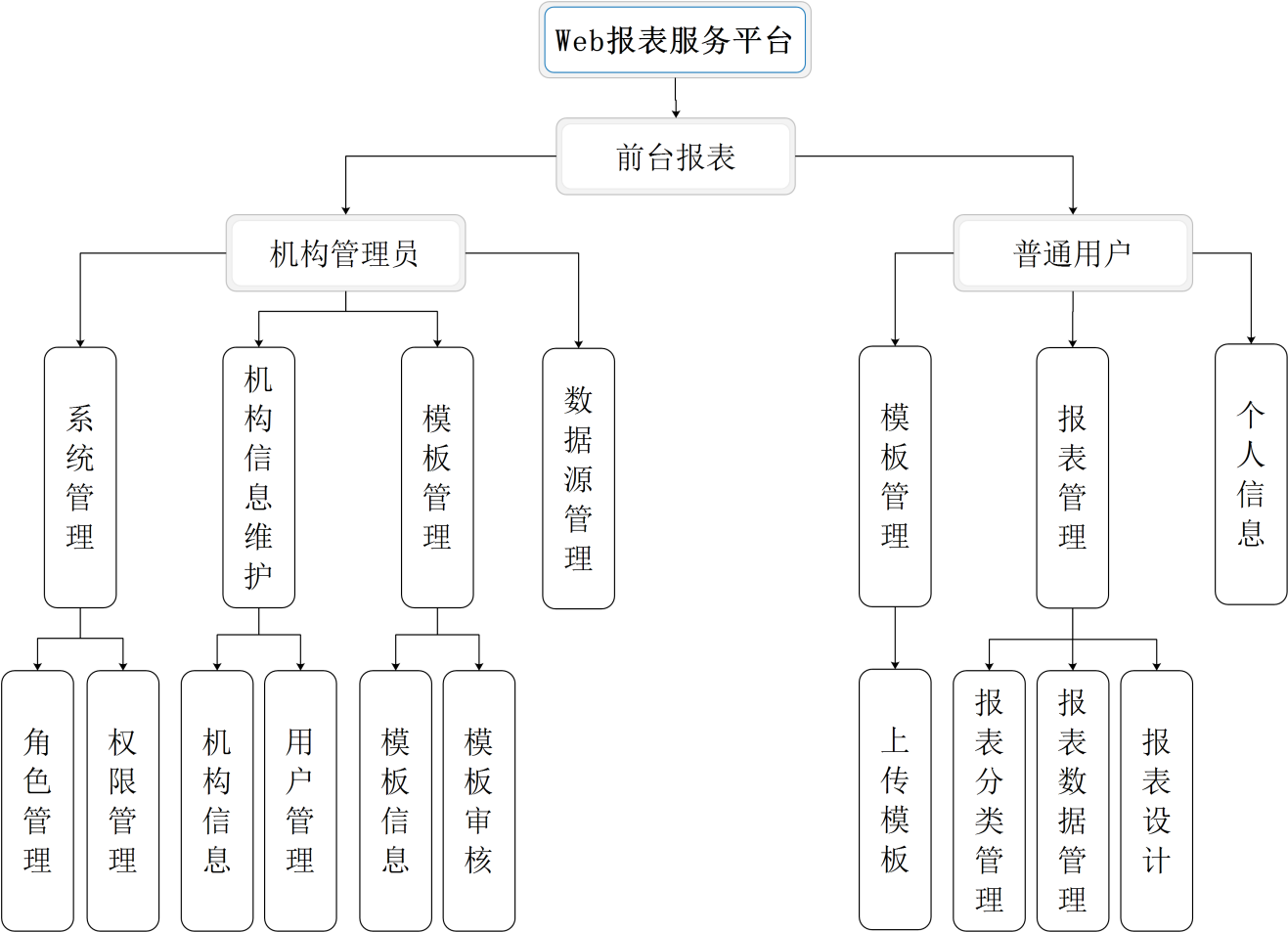
添加对taotao-manager-dao的依赖

添加spring的依赖

配置tomcat插件

|  |
| --- |
| <project xmlns=*"http://maven.apache.org/POM/4.0.0"* |

此处dada-rest的mapper层也是参考dada-manager-mapper层，因此修改的时候要及时更新本地仓库数据（保证数据的同步性）



dada-rest提供接口

系统管理：

com.dada.rest.system.service、com.dada.rest.system.service.impl、

com.dada.rest.system.controller

机构信息维护：

com.dada.rest.organization.service、com.dada.rest.organization.service.impl、

com.dada.rest.organization.controller

模板管理：

com.dada.rest.template.service、com.dada.rest.template.service.impl、

com.dada.rest.template.controller

数据源管理：

com.dada.rest.datasource.service、com.dada.rest.datasource.service.impl、

com.dada.rest.template.controller

报表管理：

com.dada.rest.report.service、com.dada.rest.report.service.impl、

com.dada.rest.report.controller

个人信息管理：

com.dada.rest.user.service、com.dada.rest.user.service.impl、

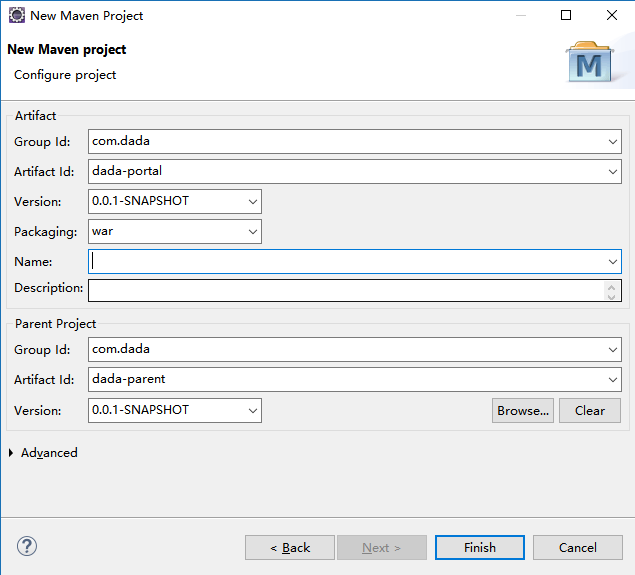
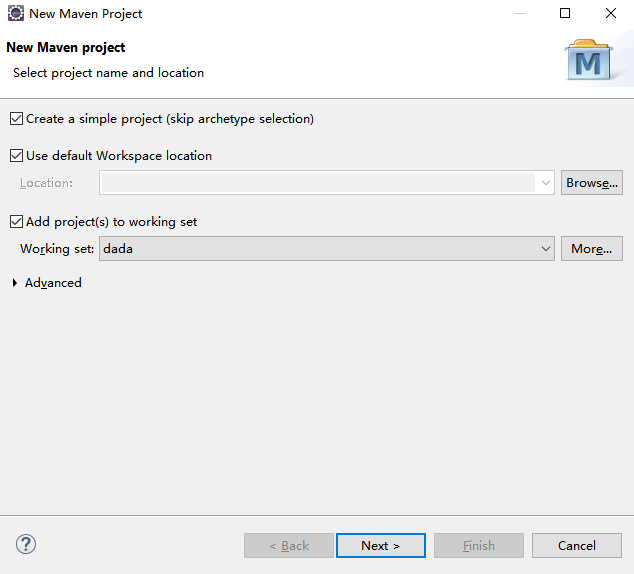
com.dada.rest.user.controller

### 3.表现层dada-portal

#### 使用的技术

* Spring
* Springmvc

#### 创建工程



#### 依赖的jar包

1. taotao-manager-pojo
2. Spring
3. Tomcat插件

Pom文件：

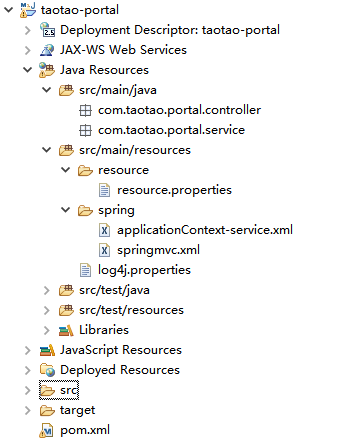
|  |
| --- |
| <project xmlns=*"http://maven.apache.org/POM/4.0.0"* |

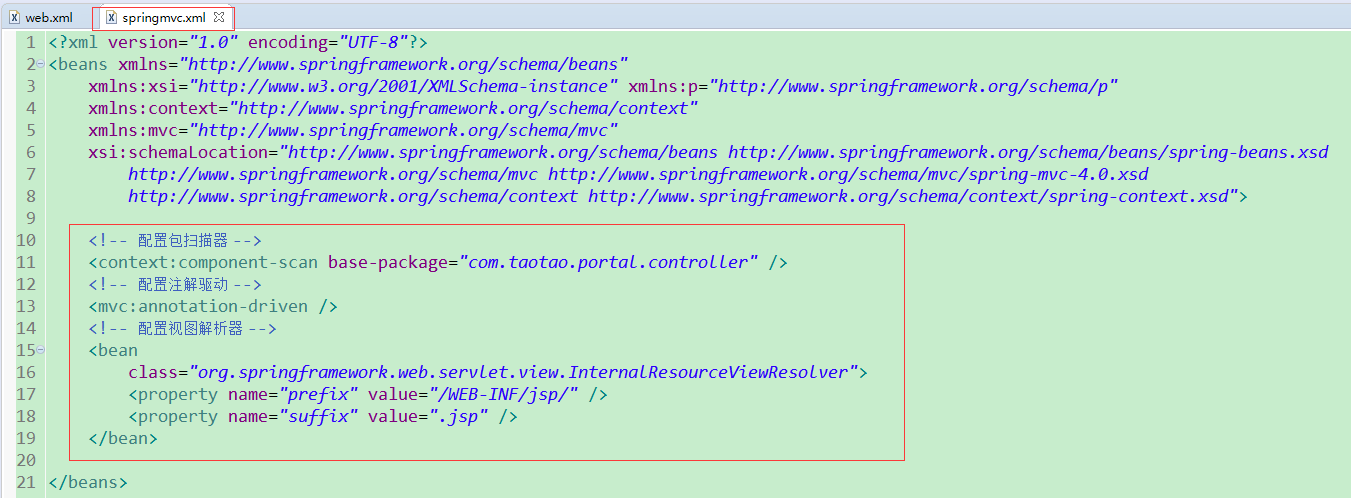
添加Web.xml文件：

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

#### 框架整合

Spring和springmvc





注释部分：启动测试时始终出现引用依赖错误，依次加载本地仓库随后再次测试

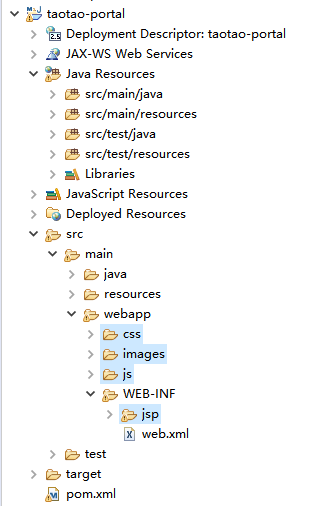
加载dada-manager进入本地仓库（过程中可能会遇到一些jar包加载问题，删除后重新导入，保证jar包导入的准确性，尤其注意一些设定仓库中没有的jar，需要自行通过工程导入或者是从其他来源下载数据进行导入）



运行过程中可能会报错，要先将相应的工程导入仓库（dada-manager-pojo、dada-manager），Maven安装工程到本地仓库，跳过测试：mvn clean install -DskipTests

#### 访问index.jsp测试

将静态文件导入到相应的webapp文件夹下



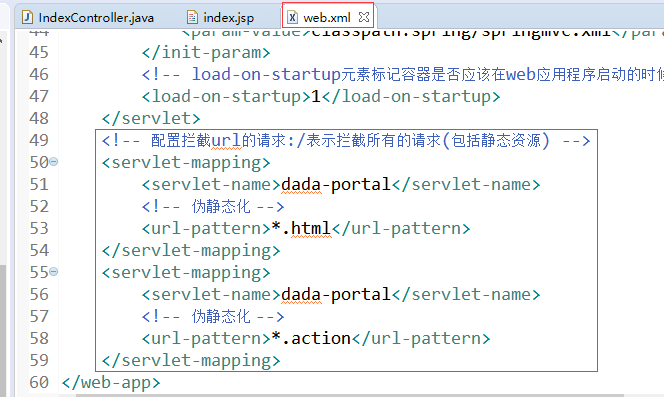
创建一个Controller，当访问http://localhost:8082/跳转到首页。

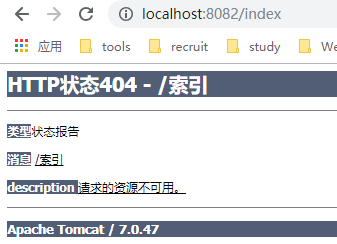
请求的url：/index

|  |
| --- |
| @Controller  **public** **class** IndexController {  @RequestMapping("/index")  **public** String showIndex() {  **return** "index";  }  } |

启动tomcat测试：clean tomcat7:run,请求路径：<http://localhost:8082/index.html>、http://localhost:8082/

http://localhost:8082/index无法访问（拦截器）（web.xml中设置路径拦截）





## Js跨域

### 什么是跨域

Js为了安全有一个限制，不允许跨域访问。

* 如果两个url的域名不同
* Url相同，端口不同也是跨域
* Ip不同也是跨域

在taotao-portal中不能使用ajax直接调用taotao-rest的服务。

### 解决跨域的问题

可以使用jsonp解决跨域的问题。

1、在js中不能跨域请求数据，js可以跨域请求js片段。

2、可以把数据包装成js片段。可以把数据使用js方法来包装，形成一条方法的调用语句。

3、可以使用ajax请求js片段，当js判断到达浏览器会被立即执行。

4、在浏览器端，先创建好回调方法，在回调方法中通过参数可以获得请求的数据。

Taotao-portal

Taotao-rest

json

Ajax请求json数据

Ajax请求js

1、请求js数据

Js，其中包含json

json

js执行，调用js方法，要求此方法已经定义。

2、响应js

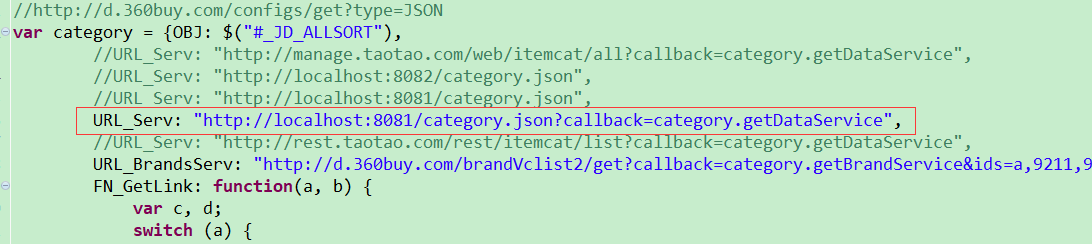
3、取出json

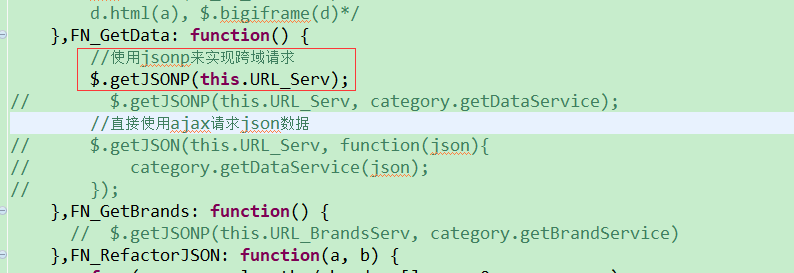
json

前期准备：

* 需要把js的回调方法先写好。（将json数据作为参数封装到某个方法）
* 做ajax请求时，需要把回调方法的方法名传递给服务端
* 服务端接收回调方法名，把数据包装好，响应给客户端









## 问题：

普通获取数据（不传数据）

// String json = HttpClientUtil.doGet(REST\_BASE\_URL + REST\_ROLE\_INFO\_ADD);

Post方式传送数据访问服务器：

//把OrderInfo转换成json

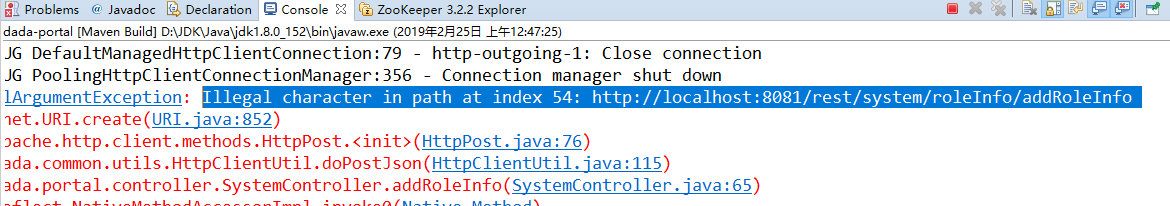
String json = JsonUtils.objectToJson(orderInfo);

// String jsonResult = HttpClientUtil.doPostJson(ORDER\_BASE\_URL + ORDER\_CREATE\_URL, json);

dada-rest实现普通的增删改查

dada-portal根据实际的需求完成相应的数据封装，调用rest服务即可

调用服务的时候报错，原因是在配置rest路径的时候在路径后方不小心添加了空格，导致系统通过HttpClientUtil请求rest服务的时候无法识别访问该路径，导致空指针异常，要细心检查文件



订单管理（创建数据参考）



问题：跨域请求返回数据报406错误：

参考链接：https://blog.csdn.net/happyljw/article/details/78350877

FastJSON、JsonUtils相应转化数据格式如下：

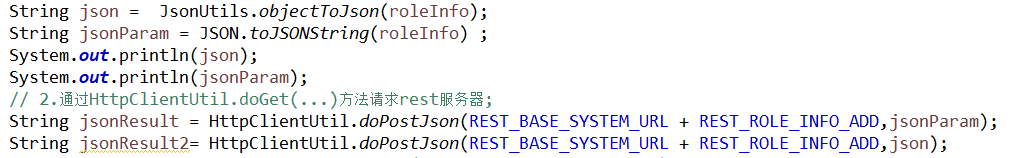
String json = JsonUtils.*objectToJson*(roleInfo);

String jsonParam = JSON.*toJSONString*(roleInfo) ;

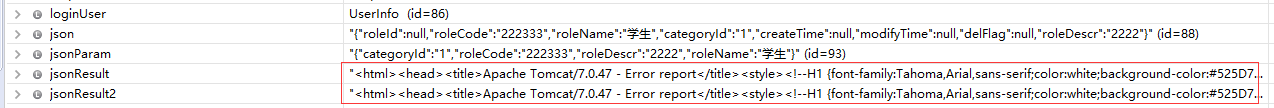
// 通过HttpClientUtil.doGet(...)方法请求rest服务器;

String jsonResult = HttpClientUtil.*doPostJson*(REST\_BASE\_SYSTEM\_URL + REST\_ROLE\_INFO\_ADD,jsonParam);

String jsonResult2= HttpClientUtil.*doPostJson*(REST\_BASE\_SYSTEM\_URL + REST\_ROLE\_INFO\_ADD,json);

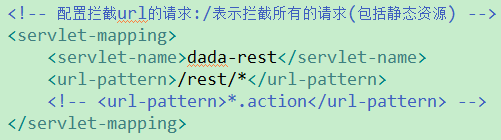


Debug测试结果显示，由JsonUtils工具类转化的json字符串为一个实体的数据（封装为json对象），会相应将没有数值的参数值设置为null；而通过JSON.toJSONString方法只是将前端页面传递的参数转化为相应的json字符串（只封装已有的属性）

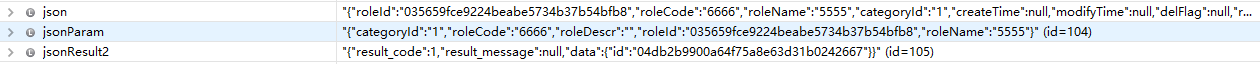


测试请求rest服务的时候报错，需要参考出错的日志信息，随后根据相应的内容进行调整：

例如404错误则是访问路径出错，需要参考相应Controller设置的url，或者查看web.xml中拦截配置的路径是否有误，此处404访问路径错误，是由于dada-rest中web.xml配置路径拦截多加了一层目录导致访问路径出错（修改为默认拦截所有路径）



相应的其他错误均可以通过debug调试，如果测试成功则显示如下信息，相应返回数据结果



## 登录拦截器流程说明

### 借助cookie+redis缓存实现

如果没有使用redis缓存数据，则cookie存储的是用户id，当服务器请求的时候则直接通过cookie存储的用户id获取相应的用户数据，

如果使用redis缓存数据，则cookie存储random随机id，在服务器请求获取数据的时候通过存储的cookie数据获取相应的redis服务器中存储的用户数据，

### 借助cookie+session实现

用户登录的时候存储用户数据到session域

如果使用session存储数据，则cookie存储random随机id（作为唯一标识session的id），在服务器请求获取数据的时候通过获取存储在客户端的cookie数据（JSEESIONID）获取相应的存储在服务器中session数据