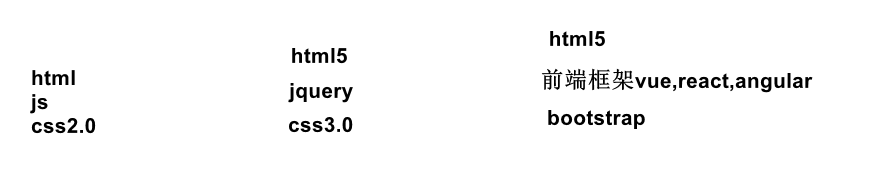
# 复习



Bootstrap是个css框架，定义了很多css

<div class=col-xs-12 col-sm-6 col-md-4>

如何实现自适应？

使用bootstrap框架

把屏幕分成超小col-xs

小屏col-sm

中屏col-md

大屏col-lg

Div class=多个样式，不同设备上生效的样式是不一样的，超小屏 col-xs生效。

Vue是个前端框架，处理标签，联网

# 用vue

## vue框架的使用

### 数据绑定

#### 效果演示

#### 分析

1. 引入vue.js框架  
   打开vue.js,报错不影响使用
2. 设置配置信息

Options={  
el:”#app”,  
data:{items:[{title:””},{}]},  
methods:{  
deleteTest:function(){}  
  
}  
}  
  
var vue=new Vue(options)

1. view标签

<div id=”app”>

<div v-for=”item in items”>

//jsp 占位符${} mybatis xml #{}

<span>{{item.title}}</span>

<span @click=”deleteTest(item.title)”>删除</span>

</div>

</div>

#### 代码

把“\亿发,移动端课前资料\vue\vue.js”拷贝到项目中

08vue.html

<!DOCTYPE html>

<html>

<head>

<meta charset=*"UTF-8"*>

<title>Insert title here</title>

<script src=*"vue.js"* type=*"text/javascript"*></script>

</head>

<body>

<div id=*"app"*>

<div v-for=*"item in items"*>

<span>{{item.title}}</span>

<span @click=*"remove(item.title)"*>删除</span>

</div>

</div>

</body>

<script type=*"text/javascript"*>

**var** options = {

el : "#app",

data : {

items : [ {

title : "item1"

}, {

title : "item2"

} ]

},

methods : {

remove : **function**(param) {

console.log("remove");

}

}

};

**var** vue = **new** Vue(options);

</script>

</html>

09vue.html

添加数据，删除数据

把值放到vue框架中

Vue框架启动时，遍历所有的input,遍历属性，发现ref 给$refs增加itemTitle属性，onblur事件绑定

Onblur事件发生后，取到input值 给$refs.itemTitle.value赋值

<input ref=itemTile>

<button v-on:click=insert>

Options{

Methods:{

Insert:function(){

//spring @autowired

//从框架中取值

Var itemTitle=this.$refs.itemTitle.value

}

}

}

<!DOCTYPE html>

<html>

<head>

<meta charset=*"UTF-8"*>

<title>Insert title here</title>

<script src=*"vue.js"* type=*"text/javascript"*></script>

</head>

<body>

<div id=*"app"*>

<input ref=*"itemTitle"*>

<!-- v-on:click相当于@click -->

<button v-on:click=*"insert()"*>

insert

</button>

<div v-for=*"item in items"*>

<span>{{item.title}}</span>

<span @click=*"remove(item.title)"*>删除</span>

</div>

</div>

</body>

<script type=*"text/javascript"*>

window.itemTitle=**new** Object();

window.itemTitle.value="test2";

console.log(window.itemTitle.value)

**var** users=["a","b","c"];

//p1:从那开始删除数据

//p2:删除几个

users.splice(1,1);//删除b

users.splice(1,1);//删除c

**var** options = {

el : "#app",

data : {

items : [ {

title : "item1"

}, {

title : "item2"

} ]

},

methods : {

remove : **function**(param) {

console.log("remove");

//比较数组中那个商品的title与param一样

**for** (**var** i=0;i<**this**.items.length;i++)

{

**var** item=**this**.items[i];

**if** (item.title==param)

{

**this**.items.splice(i,1);

}

}

},

insert:**function**()

{

//从框架中取值

//this是vue

**var** itemTitle=**this**.$refs.itemTitle.value;

**var** item={title:itemTitle};

//data:{items}

**this**.items.push(item);

}

}

};

**var** vue = **new** Vue(options);

</script>

</html>

### 部署服务器端

#### 部署数据库

1. 文件—》新建查询编辑器
2. 设置数据包的大小 ，如果不设置，sql文件态大，只能执行一部分sql

SET GLOBAL max\_allowed\_packet=1024\*1024\*400;

1. 拷贝\亿发,移动端课前资料\code\pd-web\pd.sql中的所有内容到查询窗口中，选中所有内容，执行
2. 刷新对象浏览器如图-3



图- 3

1. 看到pd\_store数据库,共13张表,如图-4。

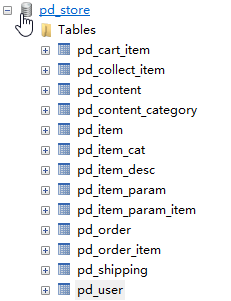
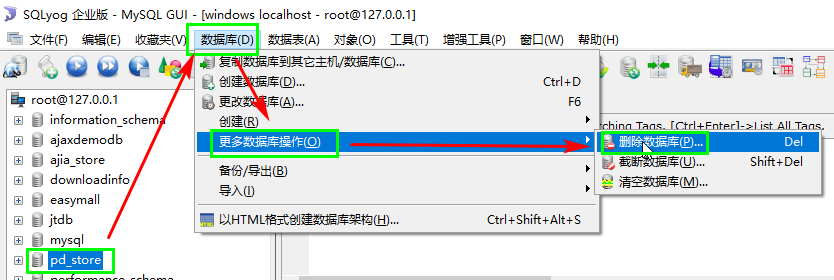


图- 4

如果不是13张表，要删除数据库，执行“SET GLOBAL max\_allowed\_packet=1024\*1024\*400;”后，再执行sql.



#### eclipse的设置

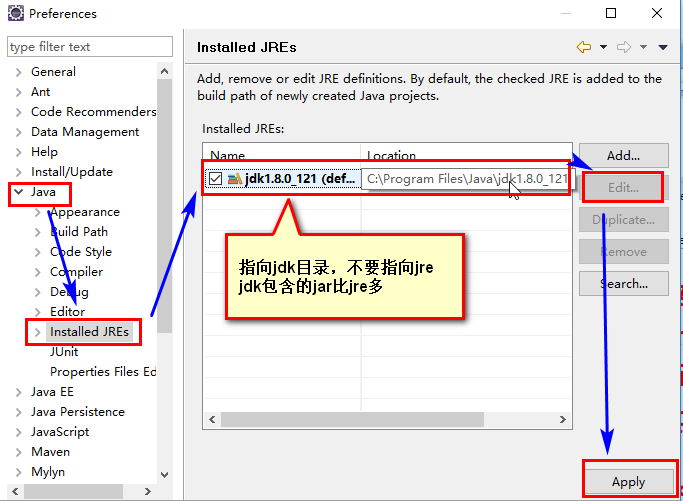
1.eclipse自动补全的设置

Windows——>Preferences——>Java-->Editor-->Content Asist，在Auto activation triggers for Java后面的文本框里只有一个“.”。现在你将其改为“.abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ”即可。

2.关闭xml验证

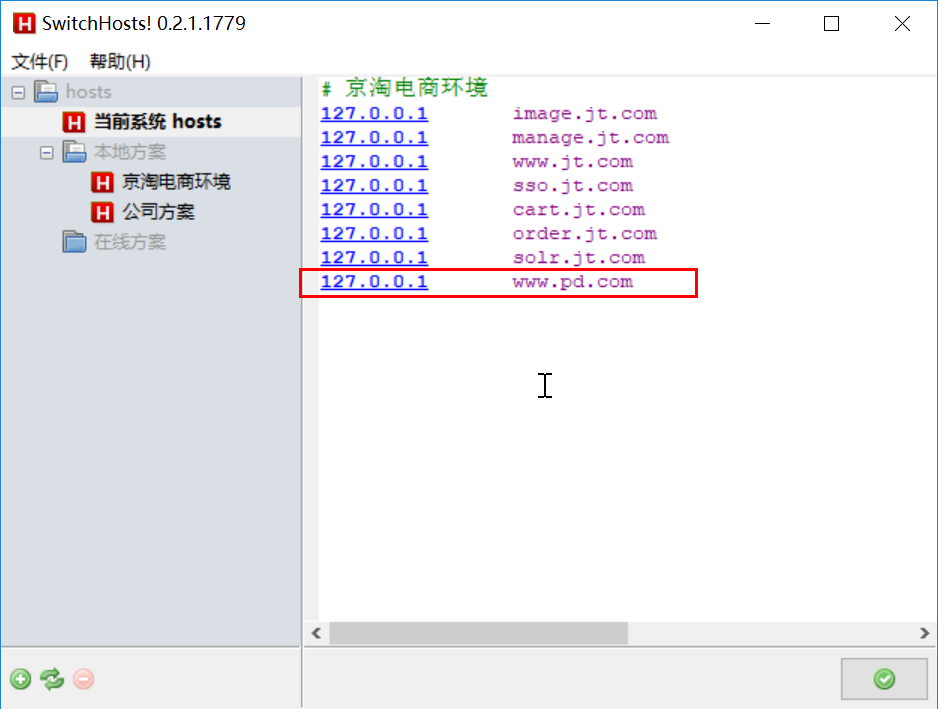
Eclipse🡪validation🡪Disable All

设置jdk



#### 添加域名

打开switchHosts,添加域名



Dos下ping [www.pd.com](http://www.pd.com)

用localhost访问注册产生的cookie，用[www.pd.com访问时不能读取cookie](http://www.pd.com访问时不能读取cookie)。

#### 部署代码

1. eclipse中import pdserver,不要用idea导入，用idea导入注册失败

代码在\亿发,移动端课前资料\code\pd-web

2.运行com.pd.RunPdAPP

3.修改MySql密码

/pd-web/src/main/resources/application.yml

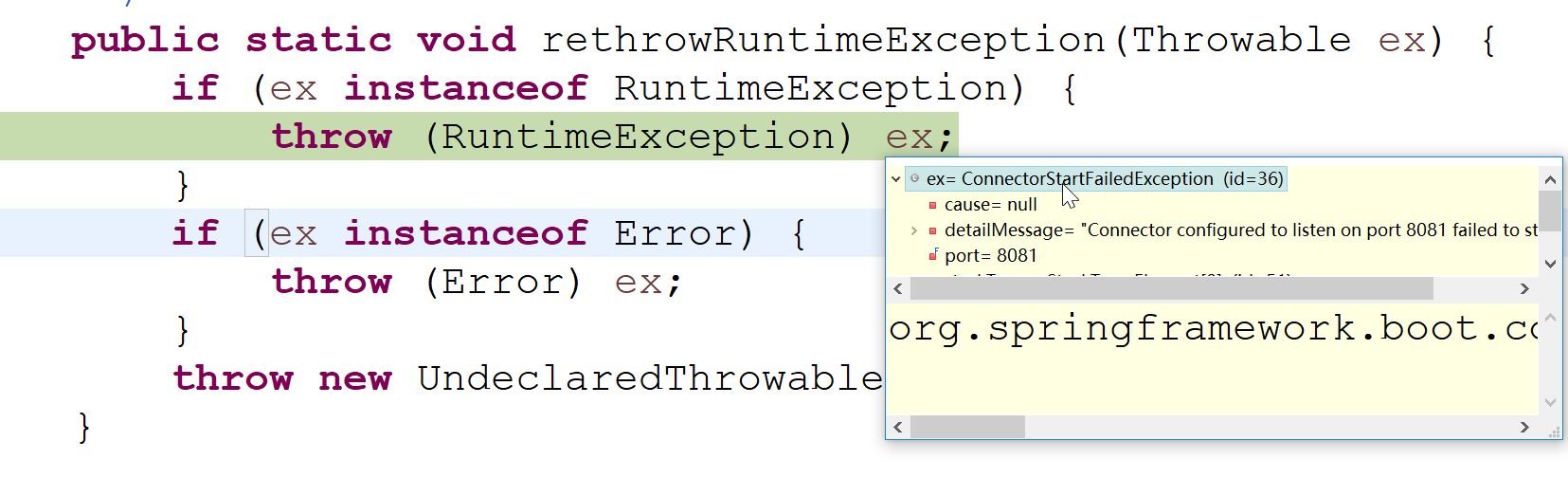
4．通过http://www.pd.com 能看到商城电脑端的首页,如图-8



图- 8

如果启动报错，eclipse中project🡪clean.

启动时要保证端口号不要被别人使用。如果端口号冲突，报以下错误



#### 添加依赖出错解决方法

1.添加依赖出错第一种解决方法：强制更新，如图- 26所示

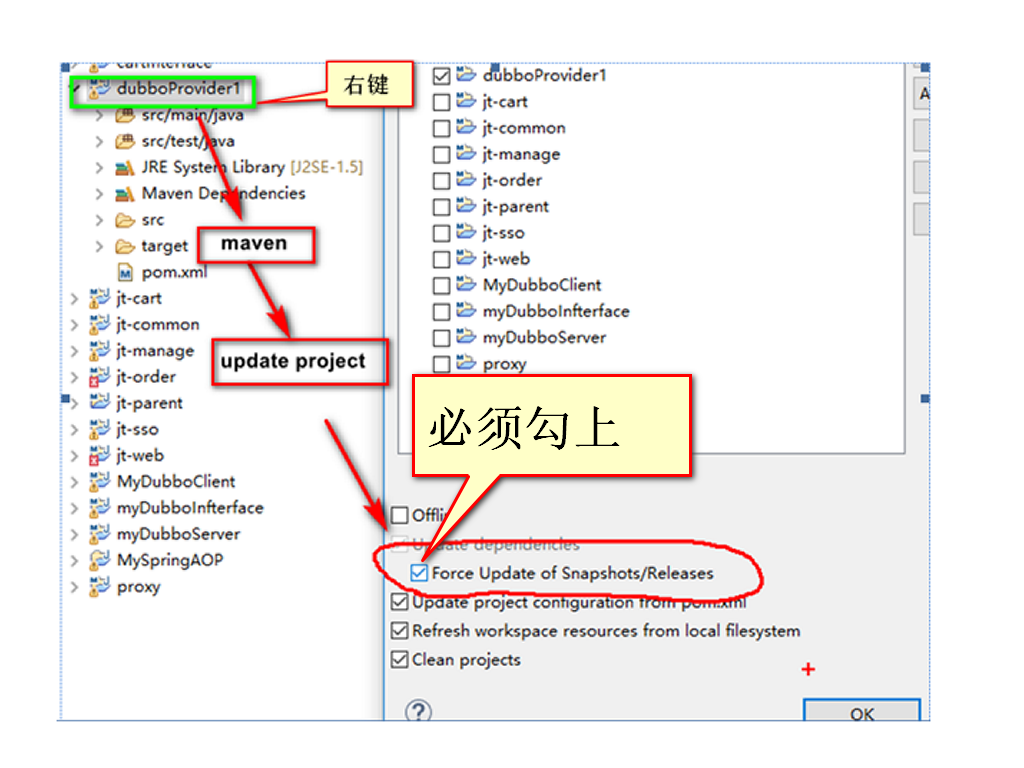


图- 26

1. 添加依赖出错第二种解决方法：

2.1通过eclipse🡪preferences查看本地仓库的位置，如图- 27所示

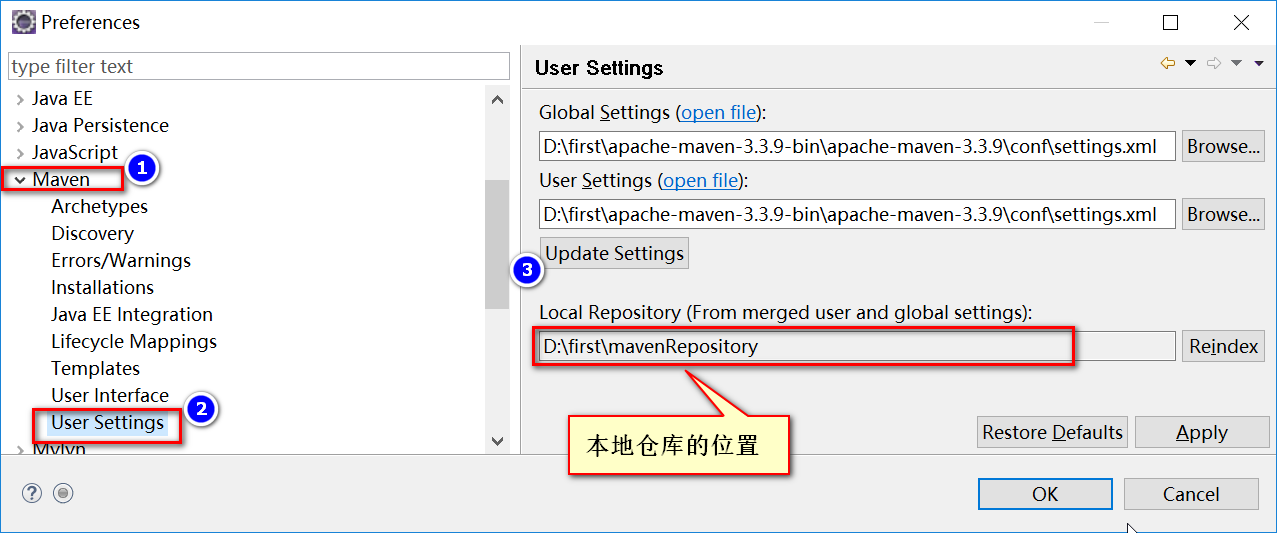


图- 27

先备份自己的仓库。

2.2解压网盘中的“mavenRepository\*\*\*\*-\*\*-\*\*.rar”，拷贝内容到本地仓库

2.3拷贝完后，本地仓库文件夹应该有org等文件夹如图-28 所示

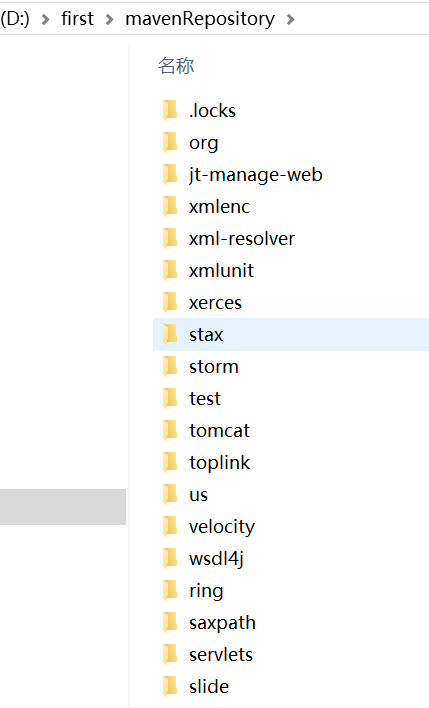


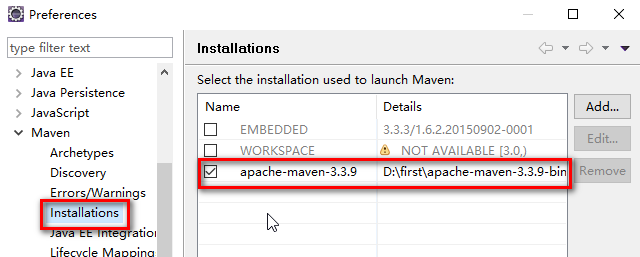
图- 28

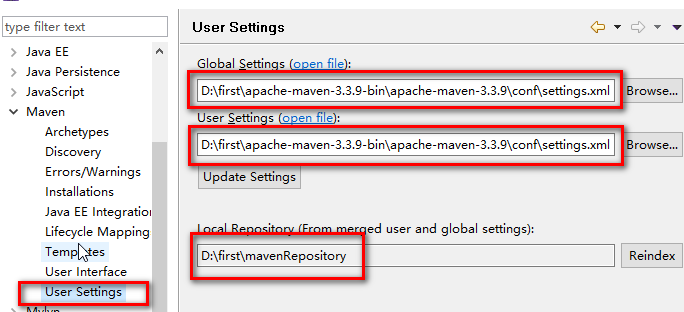
3.添加依赖出错第三种解决方法：更改工作区

尽量不要使用此种方法，因为改了工作区，要重新配置maven和jdk

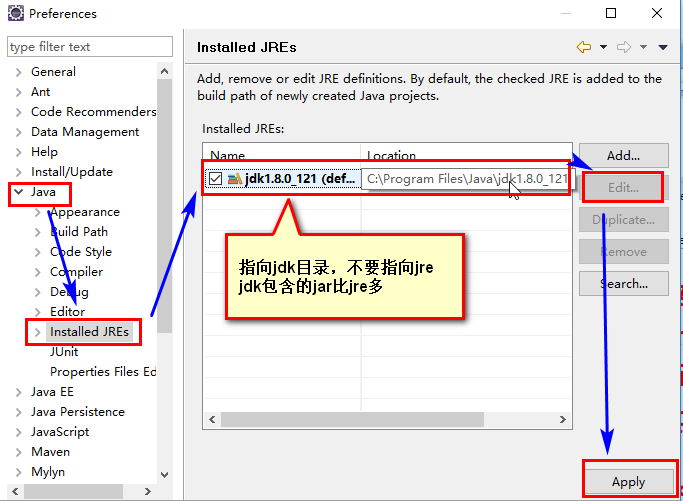
3.1Eclipse🡪file🡪switch workspace

3.2更改工作区后，要重新设置maven





3.3重新设置jdk



#### 注册帐号

在网站上注册帐号

添加地址，设置默认地址。

购物，产生订单

支付。

如果添加乱码，连接数据库时加字符集

url: jdbc:mysql://127.0.0.1:3306/pd\_store?useUnicode=true&characterEncoding=UTF-8

#### 安装jsonview

1.访问<http://www.pd.com/api/getIndexItem.html>。

2.参考“

\培优课前资料\亿发,移动端课前资料\software\jsonView\ chrome中安装jsonView.doc“安装jsonView,

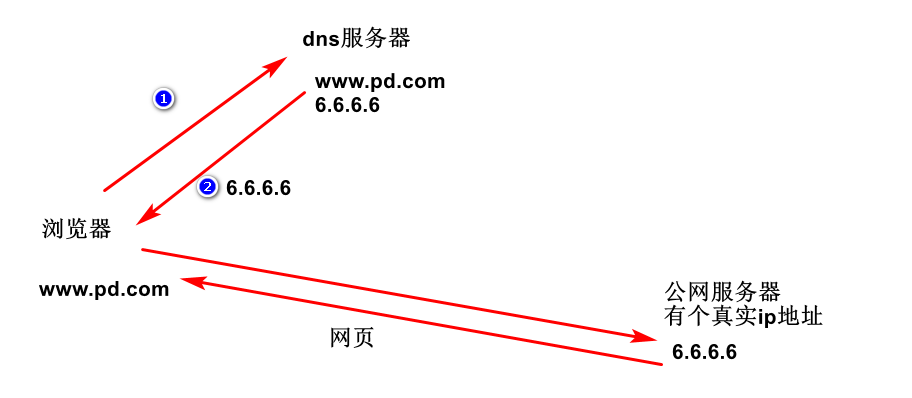
3.安装后chrome浏览器能格式化网页，如下图-9



图- 9

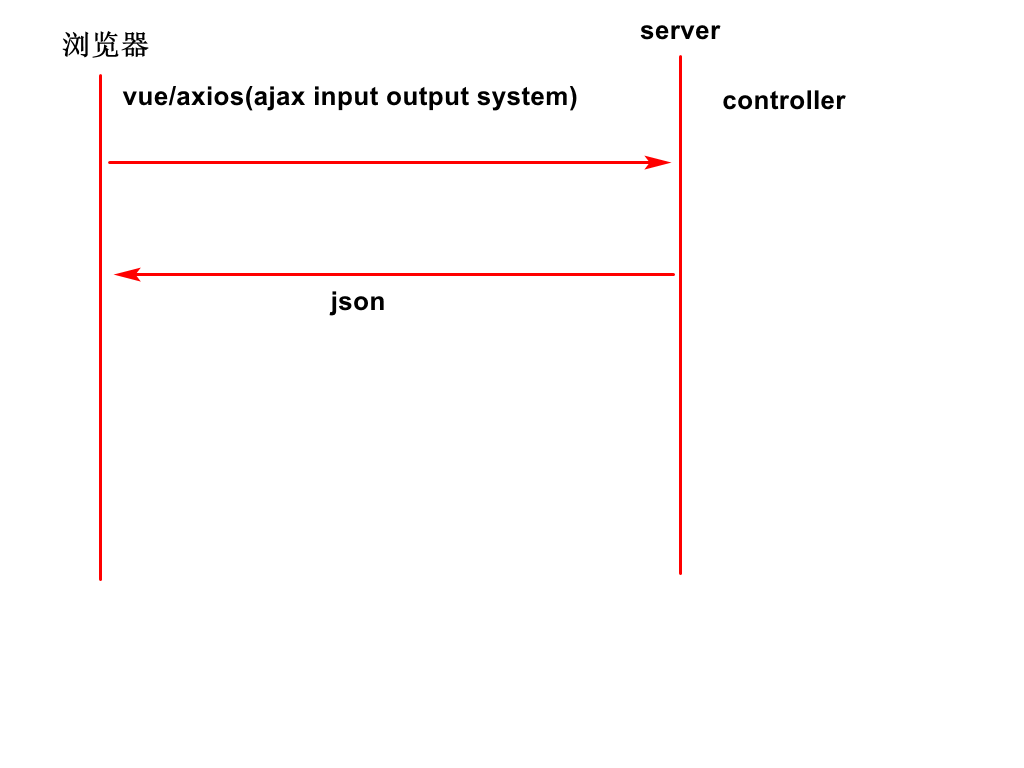
# 域名

wanwang.aliyun.com



# vue

### 联网



在pd-web中创建10.html.必须通过http访问。如果通过资源管理器访问，会产生跨域问题。

<!DOCTYPE html>

<html>

<head>

<meta charset=*"UTF-8"*>

<title>Insert title here</title>

<script src=*"../js/vue.js"* type=*"text/javascript"*></script>

<script src=*"../js/axios.min.js"* type=*"text/javascript"*></script>

</head>

<body>

<div id=*"app"*></div>

</body>

</html>

<script type=*"text/javascript"*>

**var** options={

el:"#app",

mounted:**function**()

{

**var** url="/api/getIndexItem.html";

axios.get(url)

.then(**function**(response){

//response:{config,data,heaers...}

console.log(response);

//得到服务器的信息

//{status,msg,data}

**var** serverData=response.data;

console.log(serverData);

//得到商品信息

**var** items=serverData.data;

console.log(items);

})

.**catch**(**function**(e){

console.log("出错了");

console.log(e);

});

}

}

**var** vue=**new** Vue(options);

</script>

联网成功后，取到的数据如下图-10

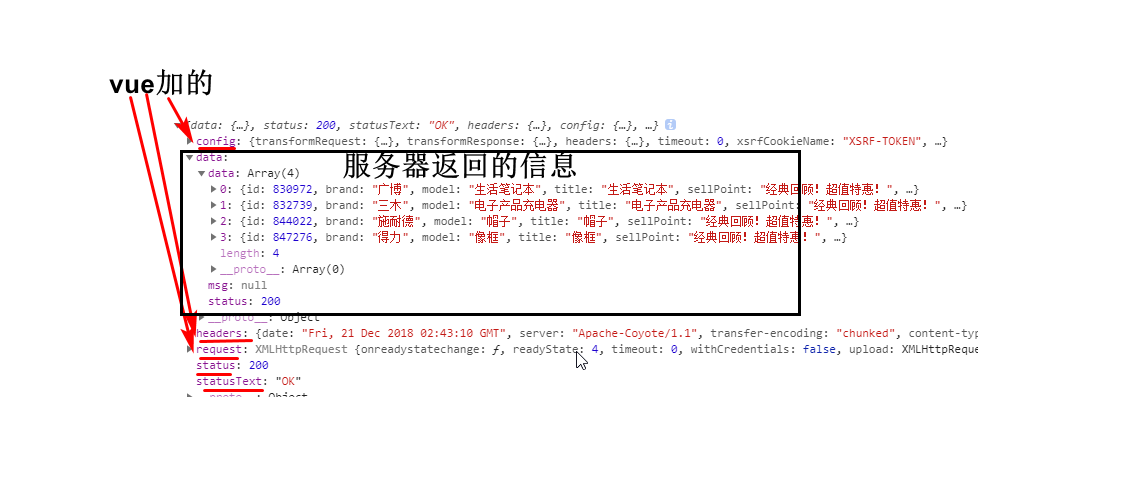
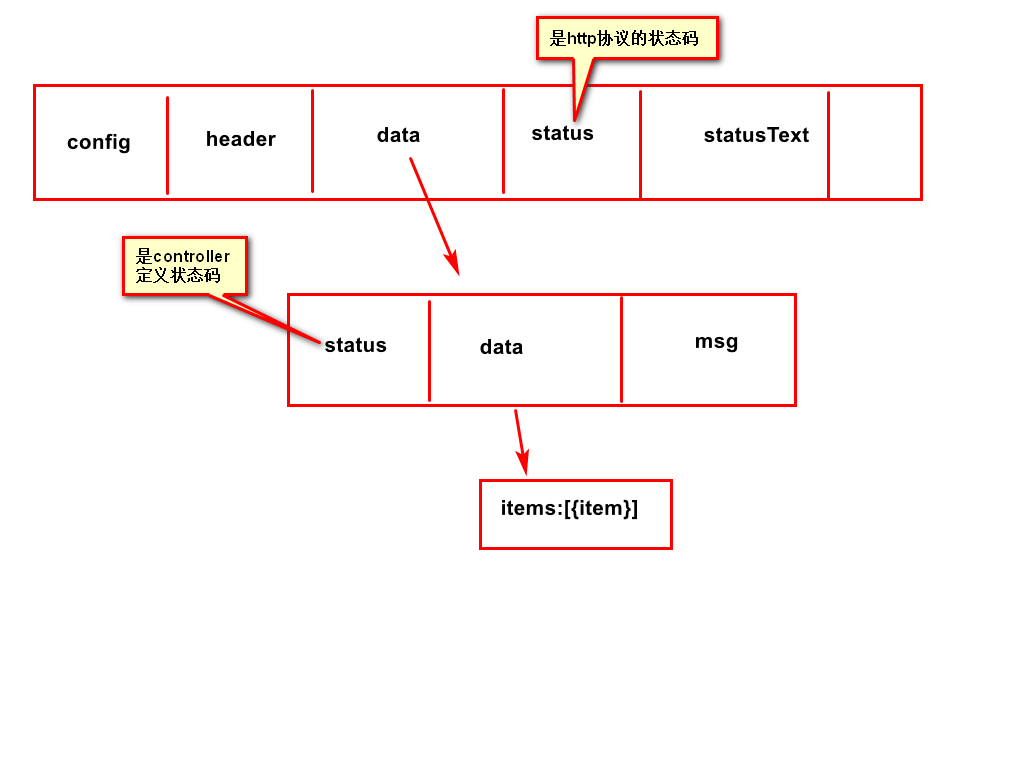


图- 10





### Vue与jquery对比

Vue的功能和jquery相似，都能实现dom操作，获取服务器端数据功能。

Jquery是个工具包。

Vue是最新框架，写的代码少，是个生态系统，有iview组件，等最新框架

企业一般用vue.

# 拼多商城网页端

## 创建响应式网页

首页包含两个row,如图-11

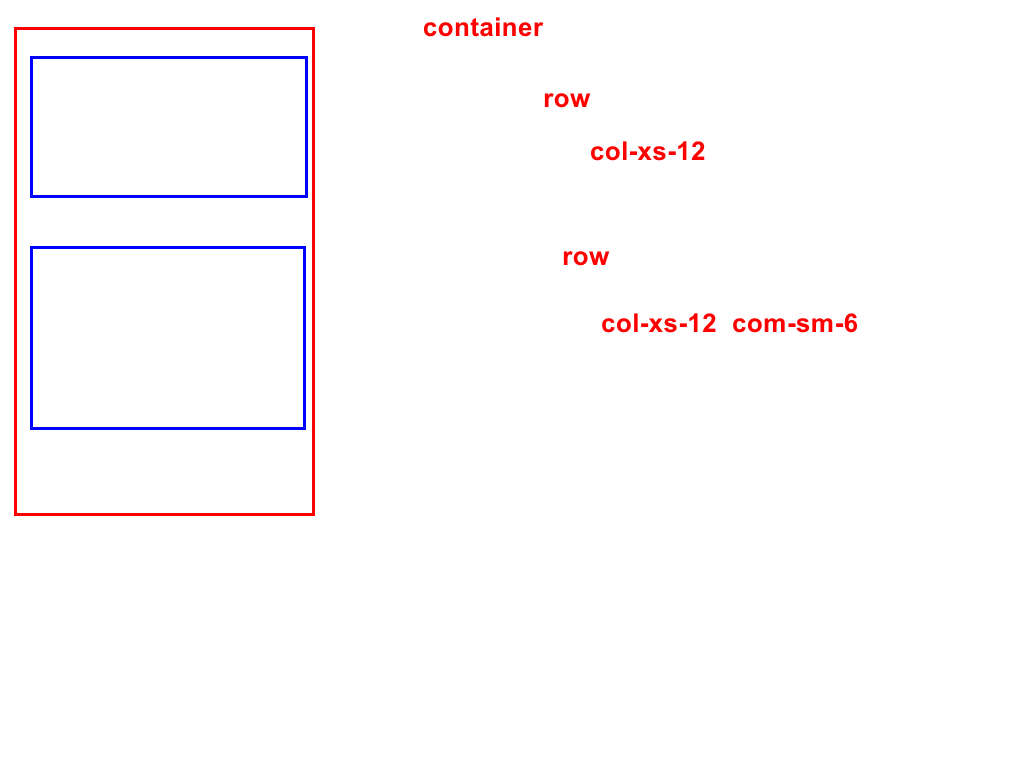


图- 11

在pd-web/src/main/webapp/pdMobile中创建网页pdIndex.html

<!DOCTYPE html>

<html>

<head>

<meta charset=*"UTF-8"*>

<title>Insert title here</title>

<link rel=*"stylesheet"* type=*"text/css"* href=*"../css/bootstrap.css"* />

</head>

<body>

<div id=*"app"* class=*"container"*>

<div class=*"row"*>

<div class=*"col-xs-12"*>

<img src=*"../images/mobileImage/banner.png"* class=*"img-responsive"* /> <br />

</div>

</div>

<div class=*"row"*>

<div class=*"col-xs-12 col-sm-6"*></div>

</div>

</div>

</body>

</html>

## 数据绑定

### 分析

界面使用div

### 实现

<script type=*"text/javascript"* src=*"../js/vue.js"*></script>

<body>

<div id=*"app"* class=*"container"*>

<div class=*"row"*>

<div class=*"col-xs-12 col-sm-6"* v-for=*"item in items"*>

<span>{{item.title}}</span><br />

<span>{{item.price}}</span>

</div>

</div>

</div>

</body>

</html>

<script type=*"text/javascript"*>

//必须放在</html>后面

**var** vue = **new** Vue({

el : "#app",

mounted : **function**() {

console.log("网页加载完成")

},

data : {

items : [

{

title : "手机",

price : 2000

},

{

title : "电脑",

price : 3000

}

]

}

});

</script>

## 联网

联网过程如图-12

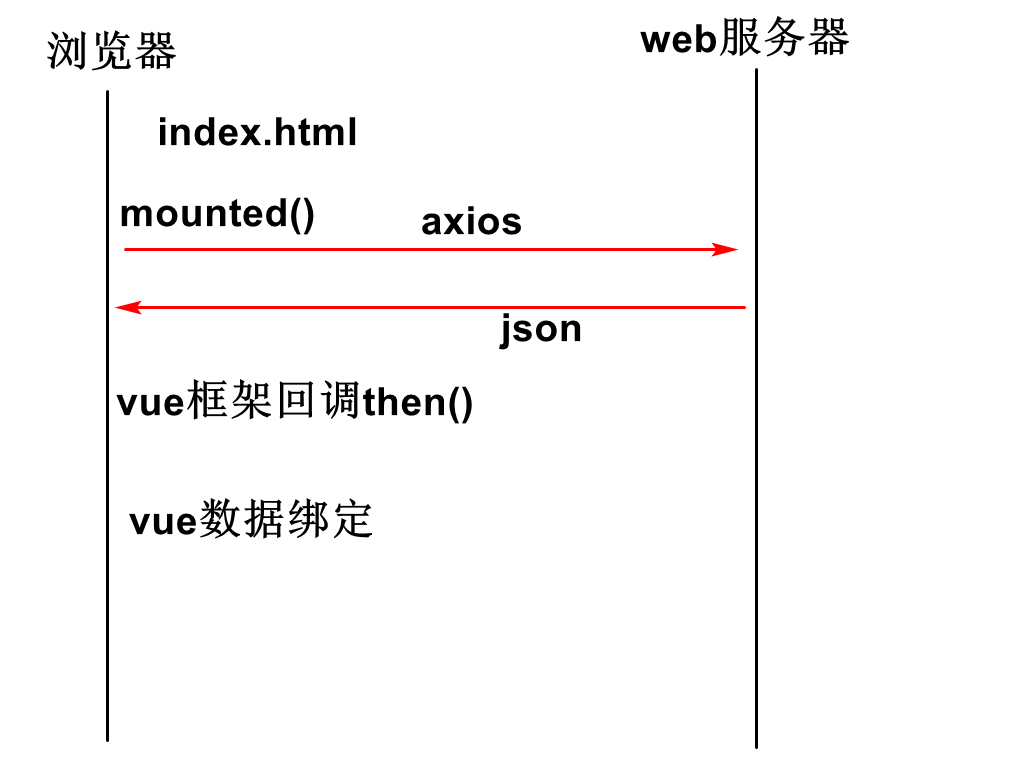


图- 12

引入js

<script type="text/javascript" src="../js/axios.min.js">

</script>

mounted : **function**() {

console.log("网页加载完成");

**var** url="/api/getIndexItem.html";

axios.get(url)

.then(

**function**(response)

{

console.log(response);

}

)

.**catch**(

**function**(err)

{

console.log(err)

}

);

},

联网成功后，取到的数据如图-13



图- 13

## 显示数据

### 显示商品信息

<!DOCTYPE html>

<html>

<head>

<meta charset=*"UTF-8"*>

<title>Insert title here</title>

<link rel=*"stylesheet"* type=*"text/css"* href=*"../css/bootstrap.css"*>

<script src=*"../js/vue.js"* type=*"text/javascript"*></script>

<script src=*"../js/axios.min.js"* type=*"text/javascript"*></script>

</head>

<body>

<div id=*"app"* class=*"container"*>

<div class=*"row"*>

<div class=*"col-xs-12"*>

<img src=*"../images/mobileImage/banner.png"* />

</div>

</div>

<div class=*"row"*>

<div v-for=*"item in items"* class=*"col-xs-12 col-sm-6"*>

<a v-bind:href=*"item.detailUrl"*>

<img v-bind:src=*"item.image"*/>

<span>{{item.title}}</span>

<span>{{item.price}}</span>

</a>

</div>

</div>

</div>

</body>

</html>

<script type=*"text/javascript"*>

**var** options = {

el : "#app",

mounted:**function**(){

//从浏览地址栏中拷贝

**var** url="/api/getIndexItem.html";

//用axios,联网成功后回调then

axios.get(url)

.then(**function**(res){

//res包含6个对象 config,data,response,headers,status,statusText

//得到controller返回的结果

**var** serverData=res.data;

//controller返回的数据一般是放在data,还有status,msg

//then中的items不能用数据绑定显示在网页上

**var** items=serverData.data;

console.log(items);

//<a href="1808detail.html?itemId=28">

//this.vue.items是data中的items

//items中有4个对象，对象有Id,title,price,没有detailUrl

//js,可以给对象加一个属性

**for** (**var** i=0;i<items.length;i++)

{

**var** item=items[i];

//加属性

item.detailUrl="1808detail.html?itemId="+item.id;

}

**this**.vue.items=items;

})

.**catch**();

},

data : {//模拟数据

items : [ {

title : "1",

price : 99

}, {

title : "1",

price : 99

} ]

}

};

**var** vue = **new** Vue(options)

</script>

## 商品详情

联网过程如图-16

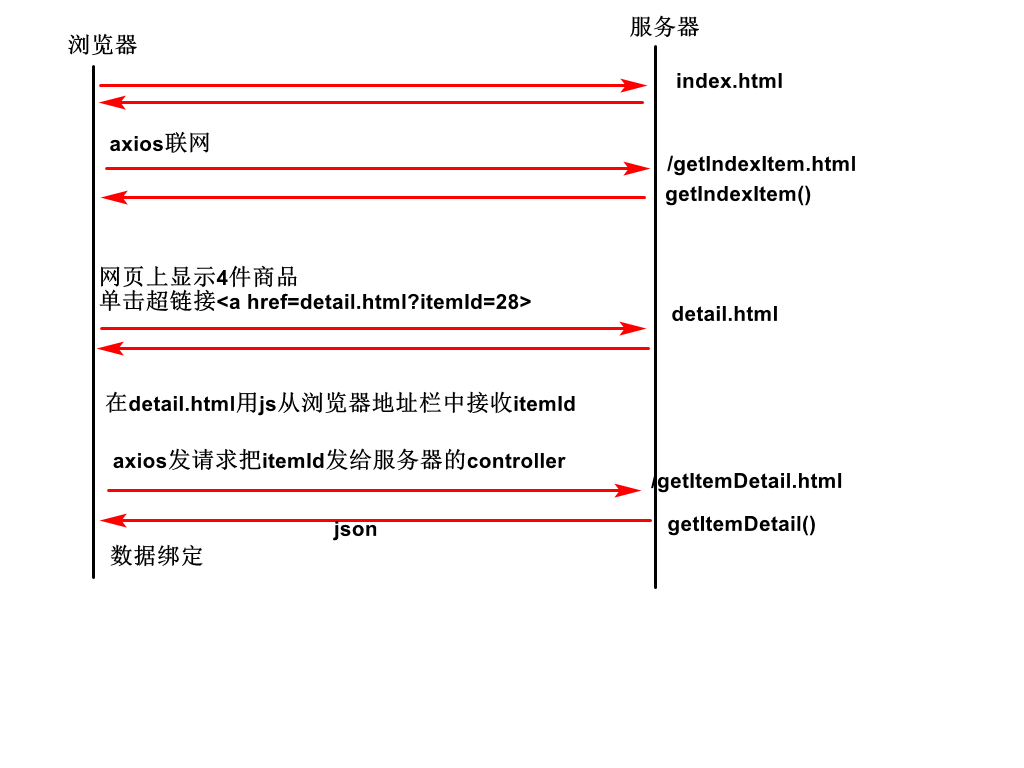


图- 16

### 响应式网页

<!DOCTYPE html>

<html>

<head>

<link rel=*"stylesheet"* type=*"text/css"* href=*"../css/bootstrap.css"* />

<meta charset=*"UTF-8"*>

<title>Insert title here</title>

</head>

<body>

<div id=*"app"* class=*"container"*>

<div class=*"row"*>

<div class=*"col-xs-12"*>

<a href=*"mobileIndex.html"*>返回</a> <br>

</div>

</div>

</div>

</body>

</html>

### 得itemId

### 显示图片

<!DOCTYPE html>

<html>

<head>

<meta charset=*"UTF-8"*>

<title>Insert title here</title>

<link href=*"../css/bootstrap.css"* rel=*"sytlesheet"* type=*"text/css"*>

<script src=*"../js/vue.js"* type=*"text/javascript"*>

</script>

<script src=*"../js/axios.min.js"* type=*"text/javascript"*>

</script>

<script src=*"../js/util.js"* type=*"text/javascript"*>

</script>

</head>

<body>

<div id=*"app"* class=*"container"*>

<div class=*"row"*>

<div class=*"col-xs-12"*>

<a href=*"1808index.html"*>返回</a>

</div>

</div>

<div class=*"row"*>

<div class=*"col-xs-12"*>

<img v-bind:src=*"item.image"* /> <span>{{item.title}}</span> <span>{{item.sellPoint}}</span>

<div v-html=*"item.desc"*></div>

</div>

</div>

</div>

</body>

</html>

<script type=*"text/javascript"*>

**var** myArray = **new** Array();

myArray["itemId"]=3;

myArray["userId"]=9;

console.log(myArray["itemId"]);

**var** options={

el:"#app",

data:{item:**null**},

mounted:**function**()

{

//.html?itemId=3&userId=9

//location.search取?itemId=3&userId=9

//split(&) [0]itemId=3 [1]userId=9

//for { split(=)}

//map itemId 3

// userID 9

//js中数组能实现map [itemId]=3 [userID]=9

**var** map=getParamMap();

**var** itemId=map.get("itemId");

**var** url="/api/getItemDetail.html?itemId="+itemId;

axios.get(url).then(**function**(res){

//res包含6个对象config,data,headers,response,status,statusText

**var** serverData=res.data;

//serverData有三个属性{status,msg,data:{image,title,sellPoint,desc}}

**var** item=serverData.data;

**this**.vue.item=item;

}).**catch**();

}

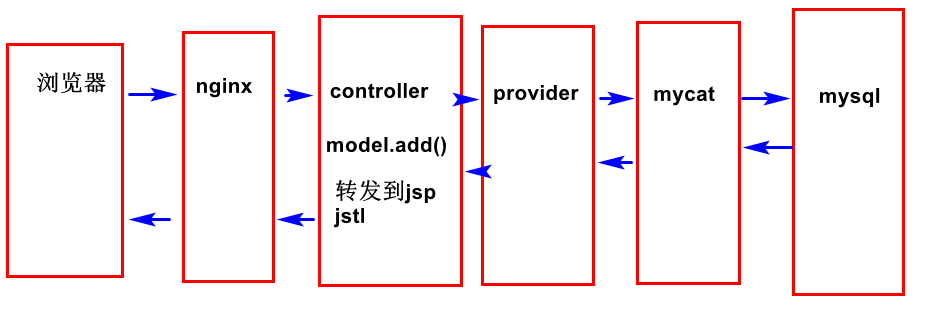
}

**var** vue=**new** Vue(options);

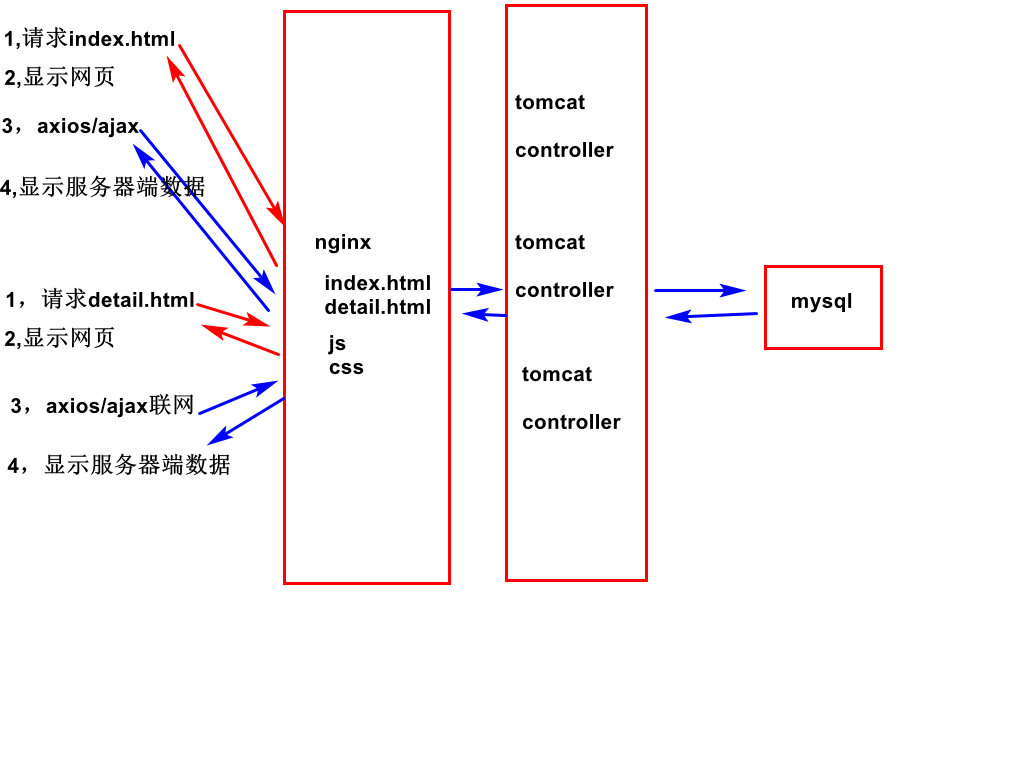
</script>

## 动静分离

动静不分



动静分离



动静分离后网页显示速度快，不用jsp了。

# 总结

