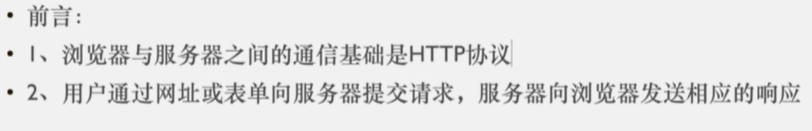
# Ajax介绍

***live server这个插件打开的文件访问本地服务器会有问题***

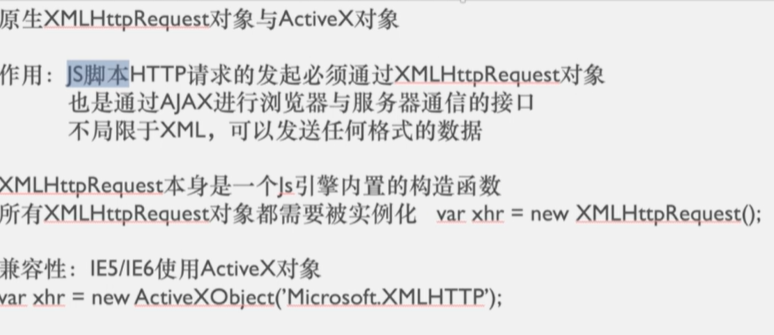
## Ajax介绍

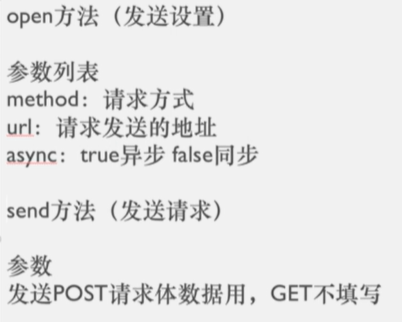
Ajax 是一种在无需重新加载整个网页的情况下，能够更新部分网页的技术。



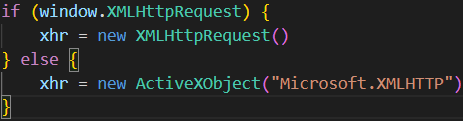


现在基本上是返回JSON数据

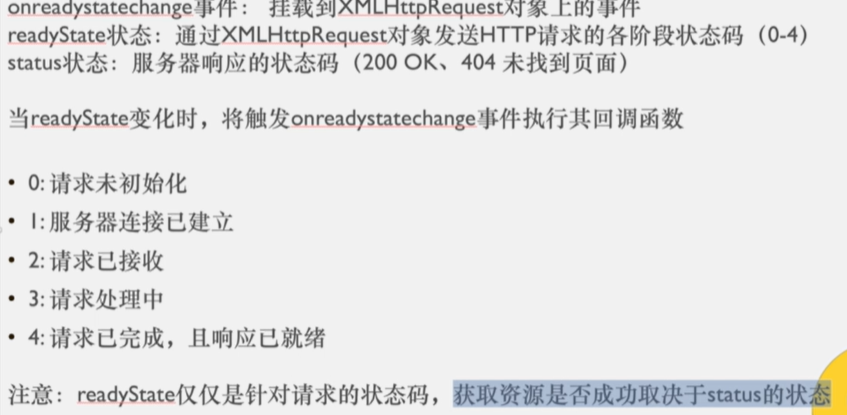


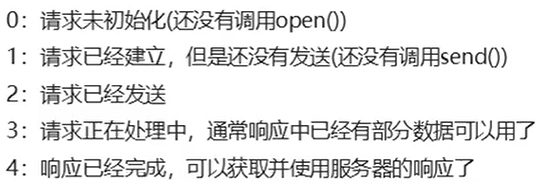


## Ajax对象的兼容性写法



## readyState状态码

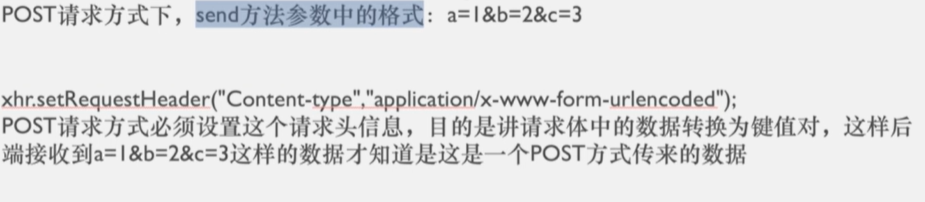




## 两种获取服务器端响应的方式的区别



## post请求设置setRequestHeader()的原因



## GET 和POST 写法

### get请求写法



### post请求写法



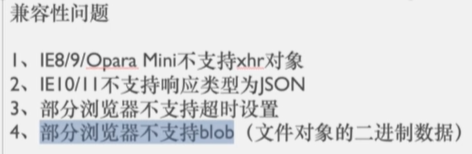
### 200~300 以及304都是成功的状态码，并不局限于200

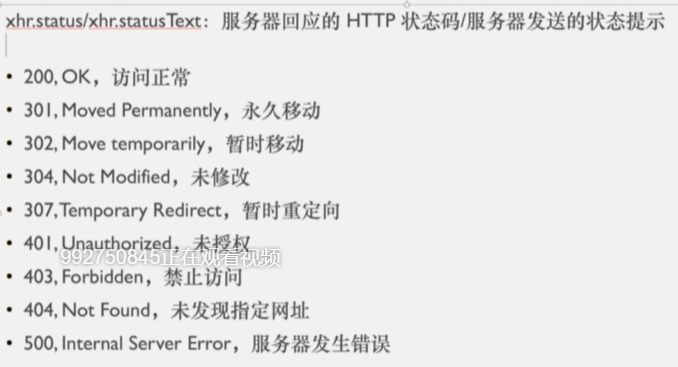


## XHR level1 和level2的区别

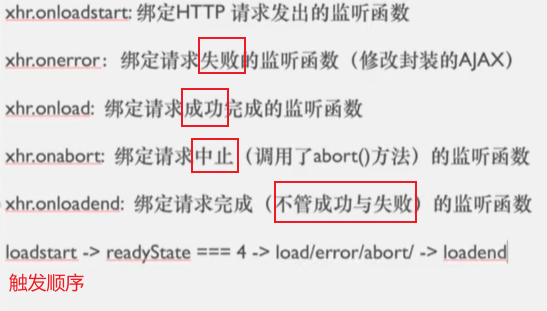
非纯文本的文件一般使用二进制传递







## Ajax五个事件 (xhr leavel2的)



onabort使用场景：请求时间过程，使用xhr.abort()终止请求

onload 请求成功触发

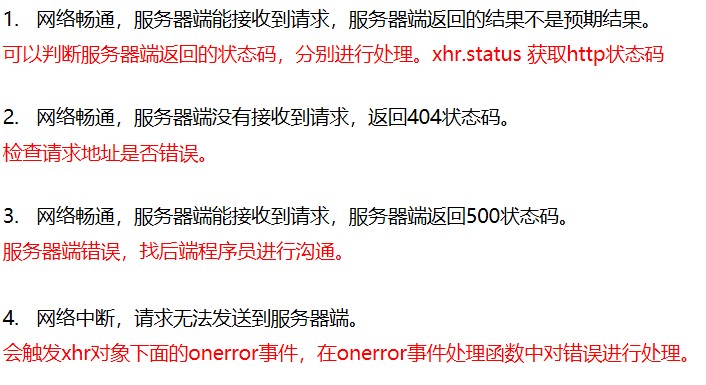
onerror 请求失败触发

onloaded在请求失败与成功都会触发，abort请求中止也会触发这个监听函数





## Ajax事件错误处理



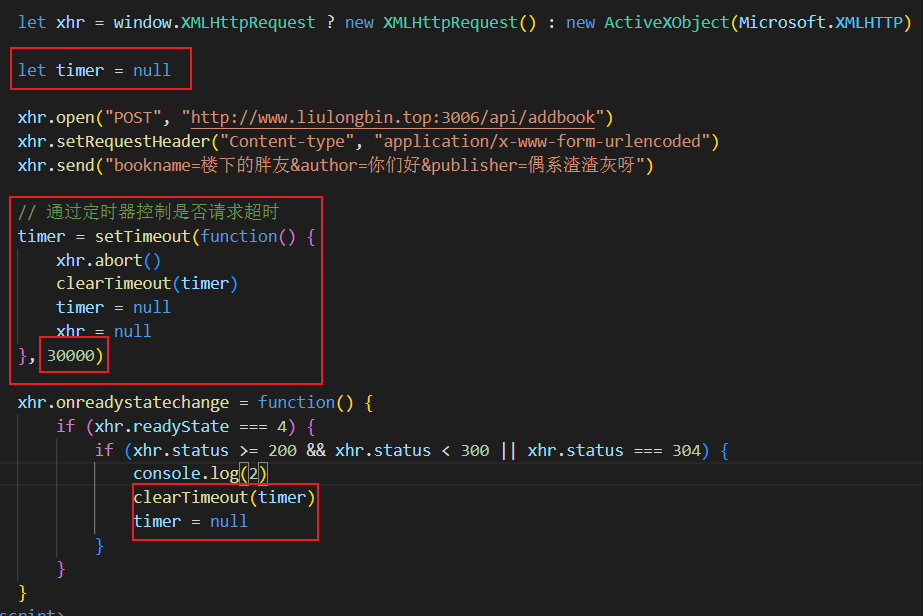
## Ajax请求超时



下面方法有兼容性问题



使用下面的方法

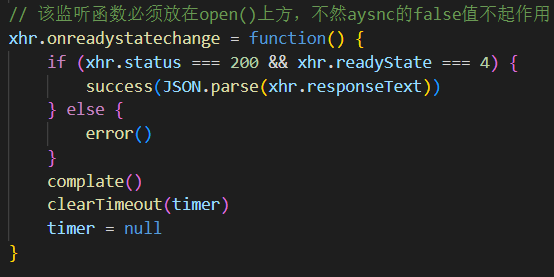


使用xhr.abort()中止请求

## Ajax中的异步和同步

async默认为true，为true时是异步的，为false时是同步的

ajax中onreadystatechange()监听函数必须放在xhr.open()上方





同步的结果是



异步的结果是（将false改为true）



将onreadystatechange()监听函数放在最下方且async为false时的结果

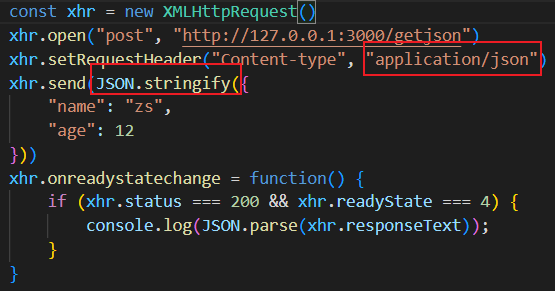




## Ajax向服务器传递不同的数据格式

### json格式数据

#### 客户端



服务器只接收字符串，所以需要将json数据转为字符串

请求头需要设置为application/json格式

#### 服务器端(nodejs)

设置处理json数据的中间件

const express = require(“express”)

const app = express()

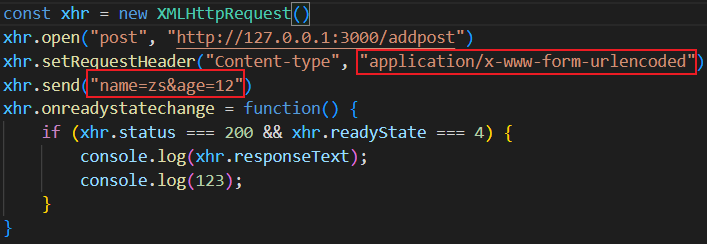
app.use(express.json())

或者

app.use(bodyParser.json())

### x-www-form-urlencoded

#### 客户端



#### 服务器端(nodejs)

设置处理json数据的中间件

const express = require(“express”)

const app = express()

app.use(express.urlencoded({extended: false}))

或者

app.use(bodyParser.urlencoded())

### 注意



## Ajax上传文件(不全)

<form …. enctype=””></form>

表单里面enctype默认是applicetion/x-www-form-urlencoded 提交纯文本数据格式的

这是一种post的提交形式，对表单数据编码, 解析成键值对的形式。

用这种方式是无法上传文件的

需要改成 multipart/form-data 以二进制的方式上传文件

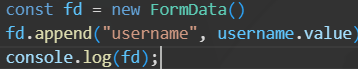
[2个半小时让你完全学会『AJAX上传视频』\_哔哩哔哩\_bilibili](https://www.bilibili.com/video/BV1Jz4y1Q7Y5?spm_id_from=333.999.0.0&vd_source=a67e813436aadc4608992466bdaea6bc) 13分钟

new FormData() 就是实例化一个表单标签对象 <form ></form>

## FormData对象

### 注意点：

直接获取FormData实例化的对象，返回的是空对象，不管是否传入数据

 结果：

获取FormData对象信息需要是通过它自身提供的方法获取



打印出FormData对象没有的属性，会返回null不是undefined

### form表单表单内容清空

原生form表单对象.reset()

### jquery的serialize()方法

将表单信息解析成 name=zs&age=12的格式

***表单必须添加name属性***



### FormData 对象的作用

1、模拟HTML表单，相当于将HTML表单映射成表单对象，自动将表单对象中的数据拼接成请求参数的格式。

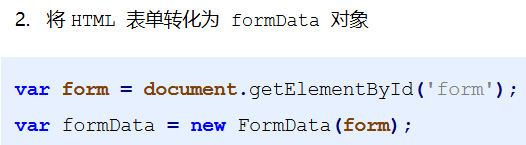
2、异步上传二进制文件

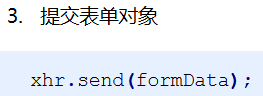
### FormData 对象的使用

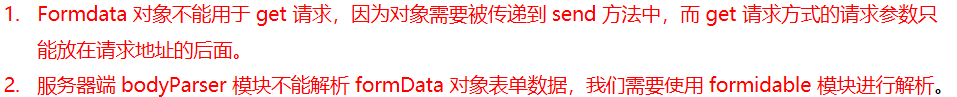
#### 客户端

使用表单一定要有***name属性***,这样才能将数据传递给后台







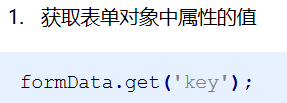


#### 服务器端

[node模块之formidable\_一个叫kevin的死肥宅的博客-CSDN博客\_formidable node](https://blog.csdn.net/weixin_43003949/article/details/106602585)

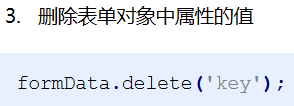


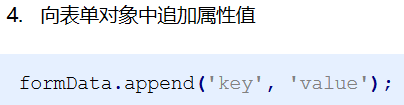
### FormData 对象的实例方法





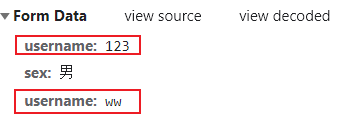








看下方，提交了两个值，但是如果服务器端不进行特殊设置的话，默认会接收后一个属性



## 上传文件

### 上传图片信息到数据库

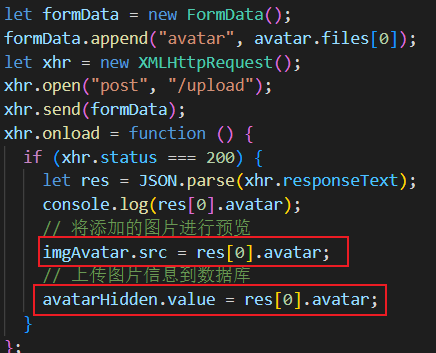
给添加图片的<input type=”file”>绑定一个change事件

当change事件触发时，上传图片到服务器。

服务器返回一个图片路径的信息

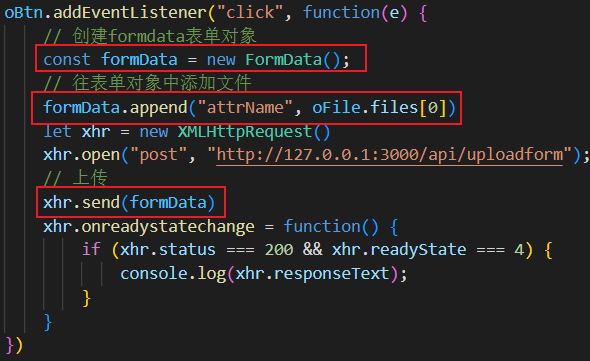
将该路径信息保存在隐藏域中 <input type=”hidden”>

然后提交表单时，将该信息连同其他的表单信息一起提交到后台数据库上

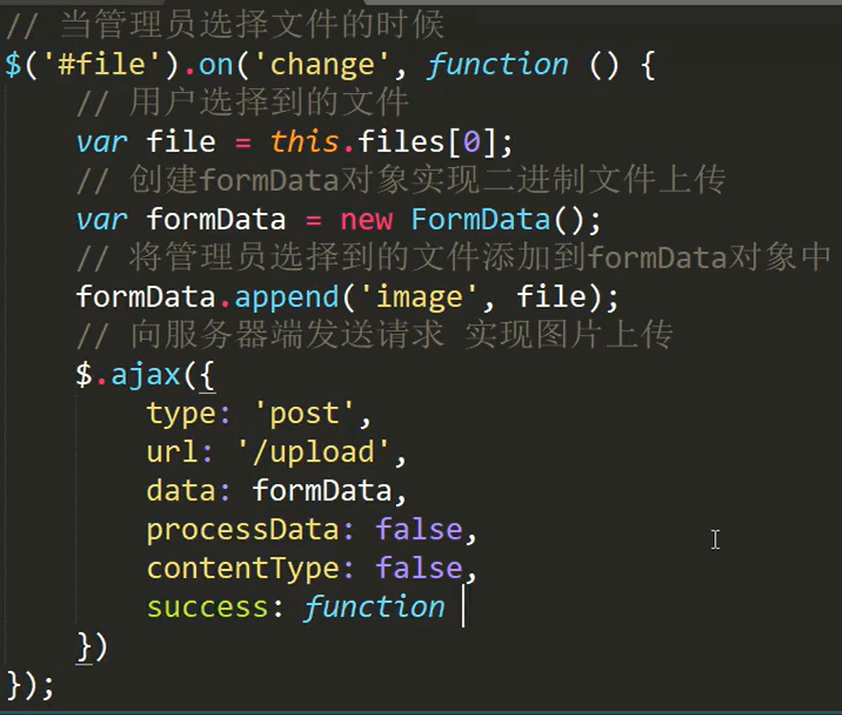


### 上传二进制文件到服务器

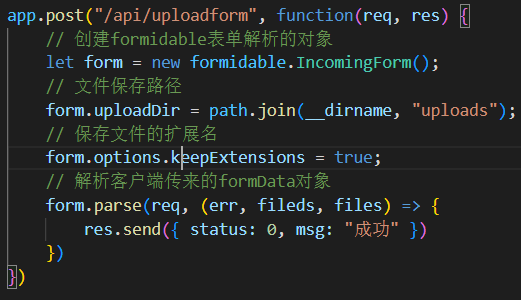
#### 客户端



##### jquery方式上传



#### 服务器端



fileds是表单提交的数据信息,属性名为name属性

files是提交的二进制文件的一些信息,比如存放路径啥的

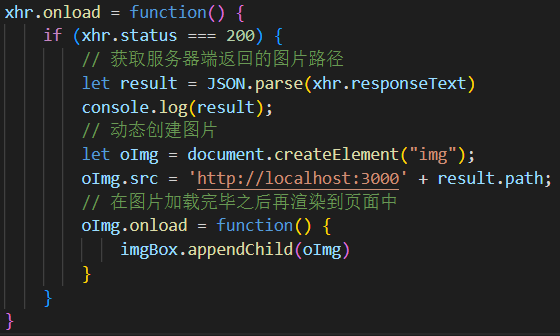
### 添加进度条



### 接收服务器端返回的上传的图片

#### 客户端代码

动态生成img



#### 服务器端



# JSONP

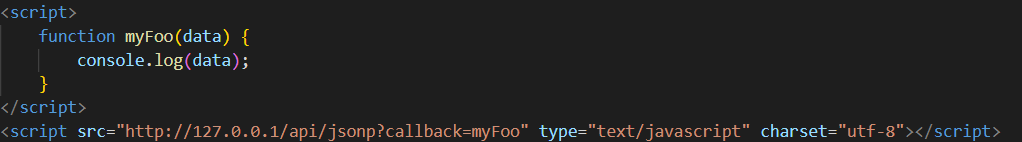
## 实现 JSONP 接口的具体代码

### 服务器端代码

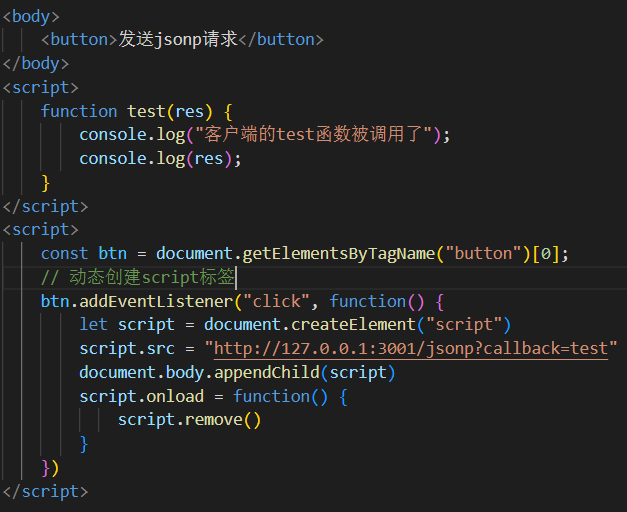


### 客户端代码

#### 第一种方式:直接通过script发起jsonp请求

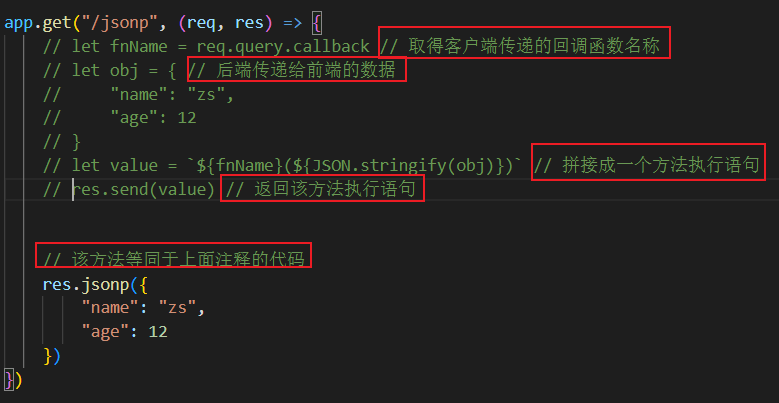


#### 第二种方式:动态创建script标签发起jsonp请求



#### 第三种:封装代码(重要)

##### 服务端代码优化



##### 前端封装

###### 执行jsonp的事件



###### 封装的函数



#### 第四种:使用jquery



# Ajax跨域携带Cookie信息

通过ajax发送跨域请求时.默认不携带cookie信息

前端发起ajax请求携带以下属性



后端配置跨域信息携带以下请求头



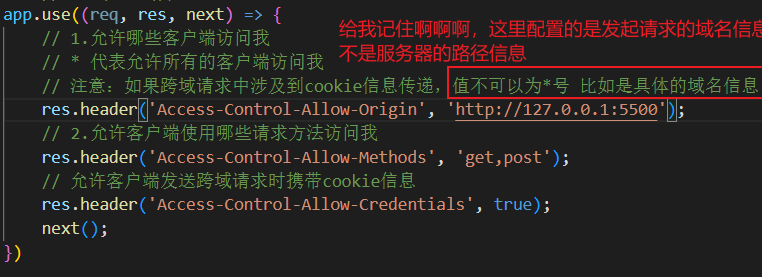
## 前端代码

登录和检测按钮的点击事件

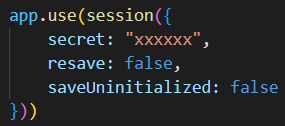


## 后端代码

### 跨域设置



### session设置



app.use(session({

secret:”xxx”,

resave:false,

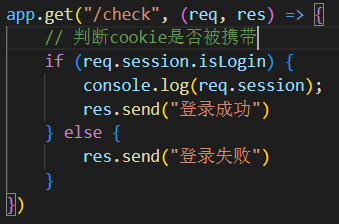
saveUninitialized:false

}))

### 登录按钮路由



### 检测按钮:通过session中自定义的isLogin属性查看cookie是否被携带



# Jquery

## Jquery发送ajax请求

### Jquery发起上传文件请求

const fd = new FormData(form)

fd.append(“avatar”, $(“file”)[0].files[0])



### Jquery发送ajax请求的loading效果设置

当网络缓慢时,可以给一个loading动画过渡(结合第三方的Nprogress使用)

jquery提供了该功能



也可以自己定义一个img标签,存放gif图片,发起请求时,显示该图片

请求结束时,隐藏图片

### 不使用jquery完成loading效果

在开始和结束位置添加对应的script

记得引入js文件









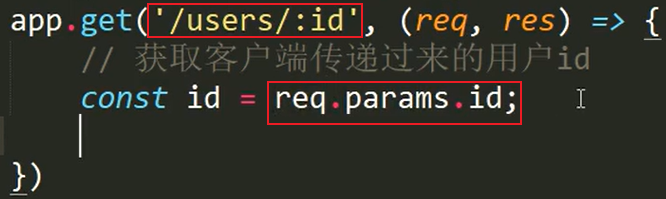
## Jquery发送jsonp请求



在使用 jQuery 发起 JSONP 请求时，如果想要自定义 JSONP 的**参数**以及**回调函数名称**，可以通过如下两个参数来指定：



# 不同请求在服务器端的接收方式



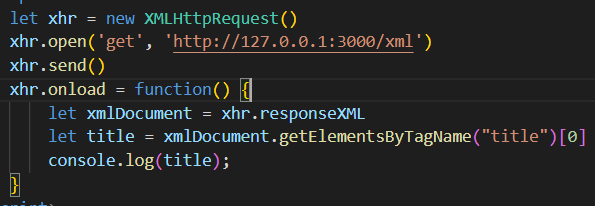
post请求中的数据

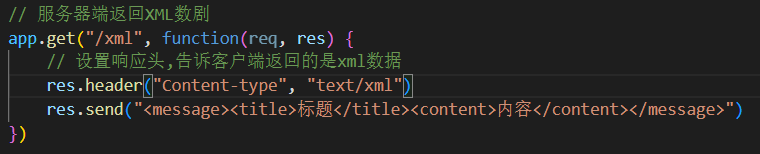
服务器端使用 req.body接收

get请求中拼接在路径后面的参数 ?name=zs&age=12

服务器端使用req.query接收

# 客户端接收xml格式数据

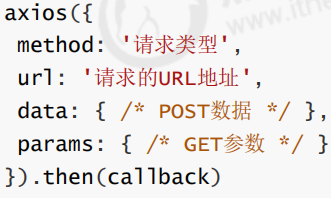




# axios请求







# 模板引擎

官方地址： https://aui.github.io/art-template/zh-cn/index.html

## 使用步骤

第二个参数接收的是一个对象，如果是数据就需要在模板中进行遍历

没有使用遍历时 不需要定义一个变量接收数据

let html = template(“tpl”, data) 直接写data

而不是

let html = template(“tpl”, { result : data})

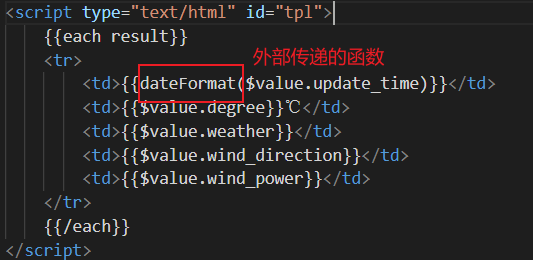




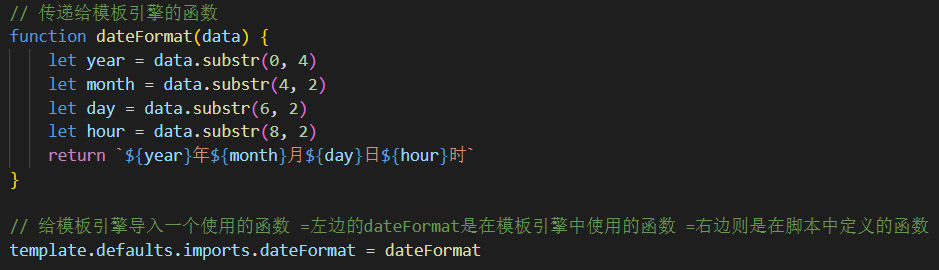
## 外部导入函数给模板引擎内部使用

注意点:template.defaults.imports.xxx= xxx 要定义在生成模板(模板与数据进行拼接)前使用,否则模板内部找不到对应的函数

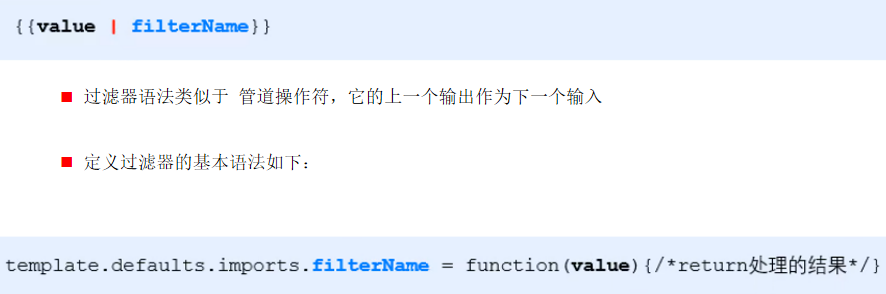
模板代码

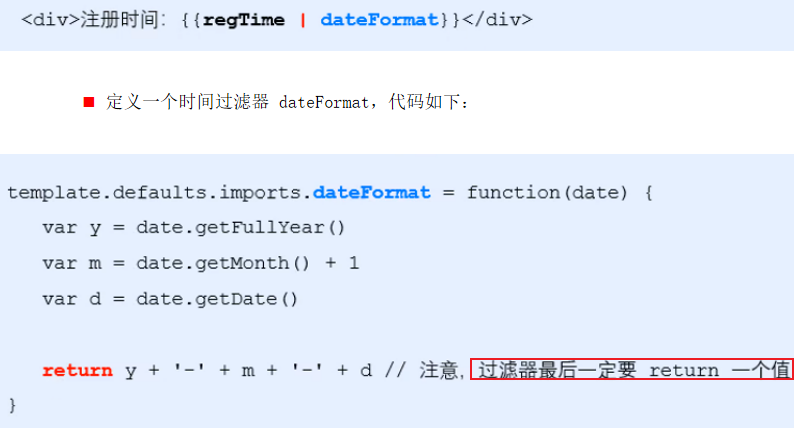


js脚本代码



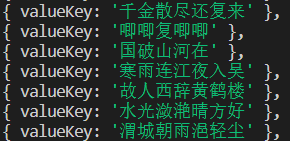
## 过滤器



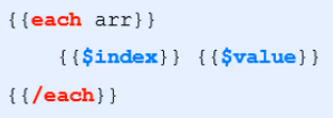


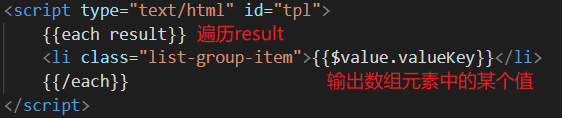
## 遍历模板

数据格式



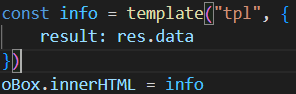
模板编写 ***{{$index}}是索引值***





也可以是对象当作数据传递给模板引擎{ {name:”ZS”,age:12}, {name:”lise’,age:22} }

模板编译

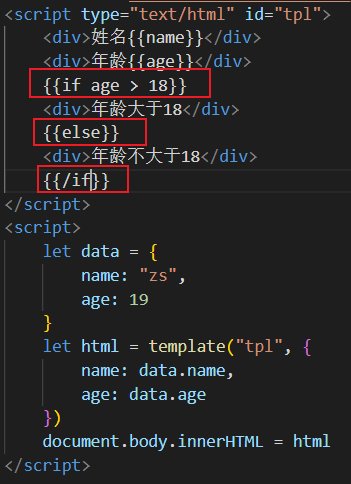


tpl：模板id

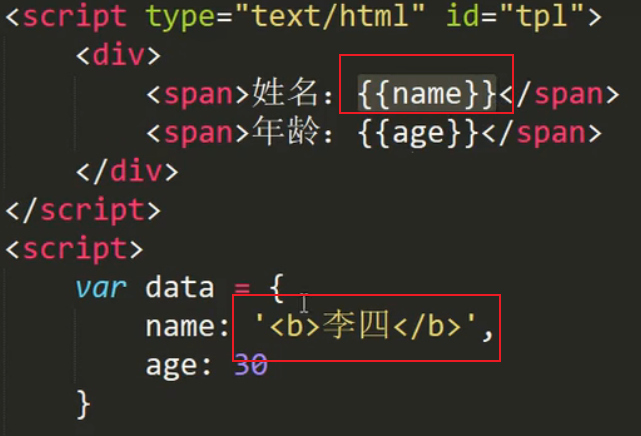
result：模板中使用的变量

res.data：传递给模板使用的数据

## 循环判断



## 输出原文信息



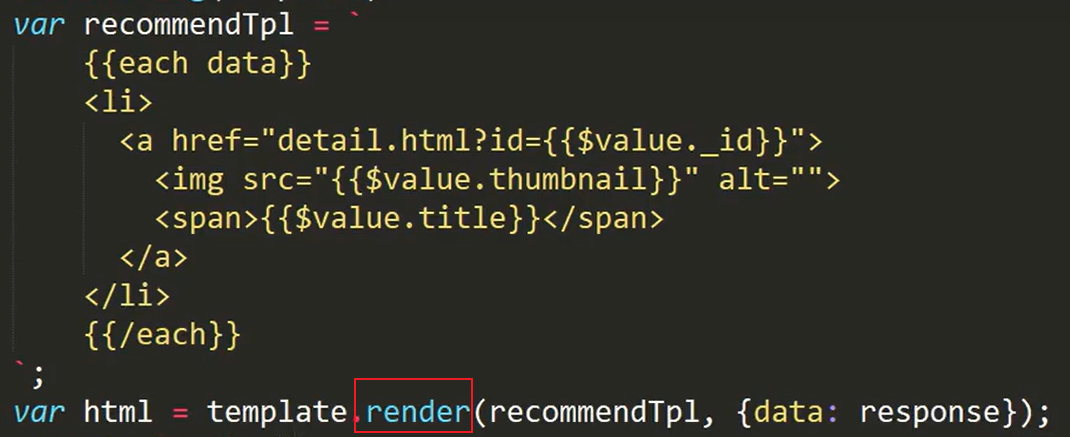
会直接输出 <b>李四</b>

在模板中的{{@ value}} 加上一个@

## 在js文件中编译模板

使用模板引擎给我们提供的render()方法

为了让模板变成公共的部分所以要写在js文件中



# 其他

## 可以触类旁通的小问题

使用自己封装的ajax去获取两个请求，但是只返回了后面一个请求的数据



原因在于

两次请求都是异步的，

两次发起请求的xhr对象是同一个

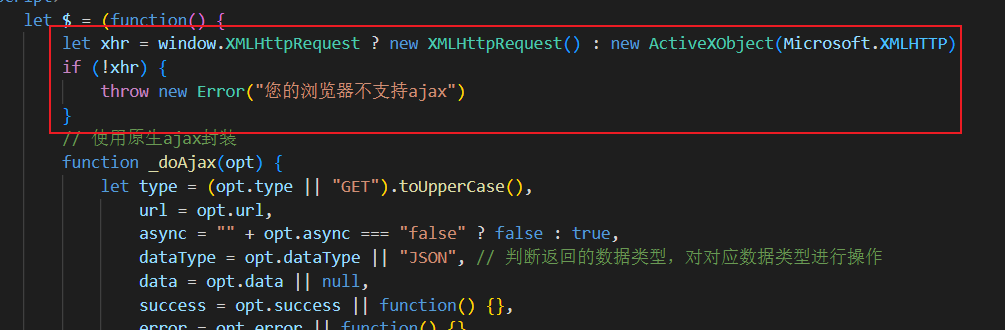
当其中一个异步请求在执行时，另外一个异步请求又发起了，对象就跑去执行另外一个异步

所以需要将该对象定义在函数\_doAjax中，这样每次发起请求的都是不同的对象

但是，如果两个都是同步的就没有这个问题、或者前面的是同步的，后面的是异步的。

不能前面是异步的，后面的请求是同步的。

**总而言之，直接放到函数体里面生成xhr对象就对了**



## 面试

[AJAX版本、响应状态、超时设置、同步与异步 (qq.com)](https://ke.qq.com/webcourse/2837895/104501696#taid=12759012104424839&vid=5285890788131646351) 一小时37分开始将面试相关的

## 低版本 IE 浏览器的缓存问题

