express

**一，Express初步**

**1，什么是Express?**

express单词的意思：快速地，迅速地

官网：<http://www.expressjs.com.cn/>

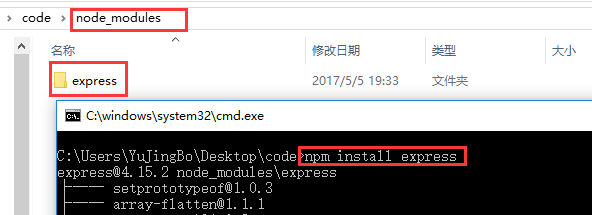
Express 基于 Node.js 平台，快速、开放、极简的 web 开发框架。

**2，安装Express**

我们可以把express理解成一个包，如何去安装一个包？

通过npm install

安装过程如下：



**3，创建第一个Express应用**

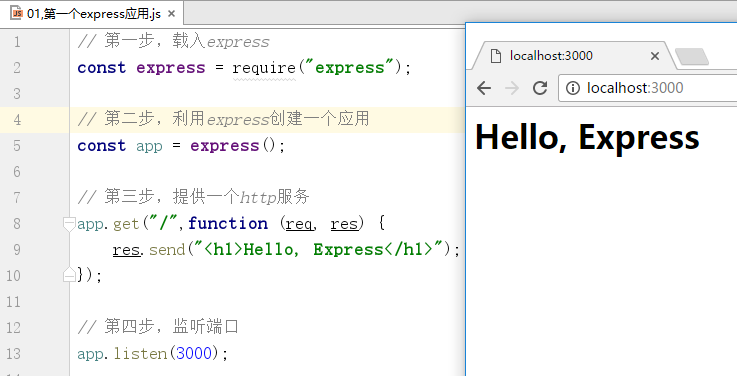
步骤：

**第一步：创建一个JS文件**

**第二步：缩写如下代码：**



**第三步，执行上面的js文件：**



**4、使用express处理表单**

**Get请求**

**Post请求**

**图片上传**

使用图片上传，我们需要使用一个中间件formidable

设置存放目录

设置大小

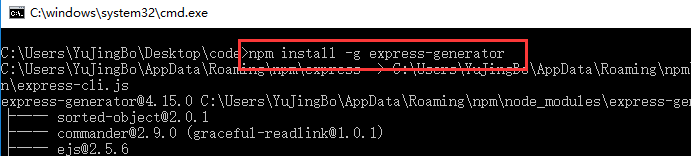
**二，应用生成器**

在一个项目中，有css文件，js文件，图片，等等一些资源，我们可以通过应用生成器，去生成一个项目的大概结构。

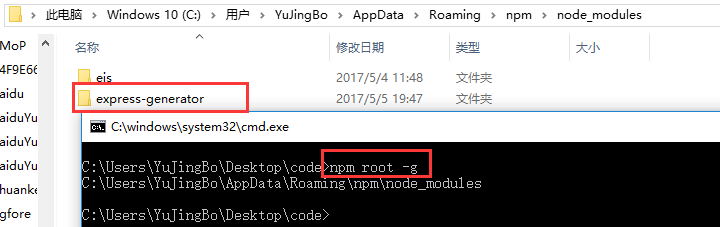
**1，安装生成器**

我们采用全局安装生成器： npm install -g express-generator

安装如下：



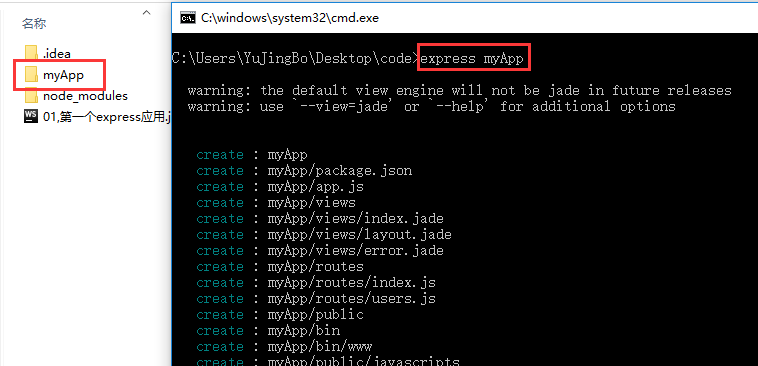
如何查看全局安装的位置： npm root -g



**2，创建应用**

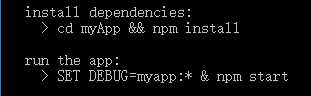
express 加上项目的名称

**（1）创建如下项目结构如下：**

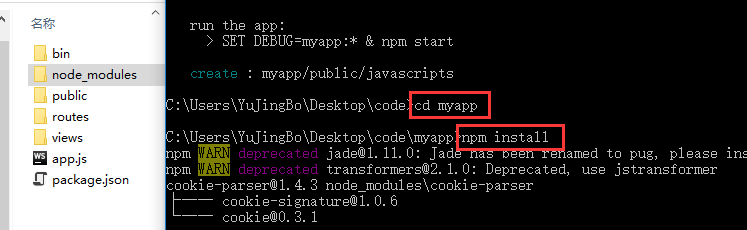


**安装依赖包：**

通过提示，我们知道，还需要执行下面两个命令：

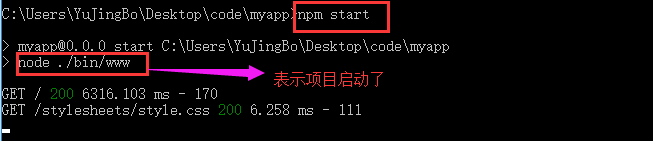


执行如下：



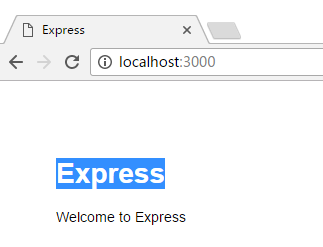
**启动项目**

通过npm start也可以启动项目：



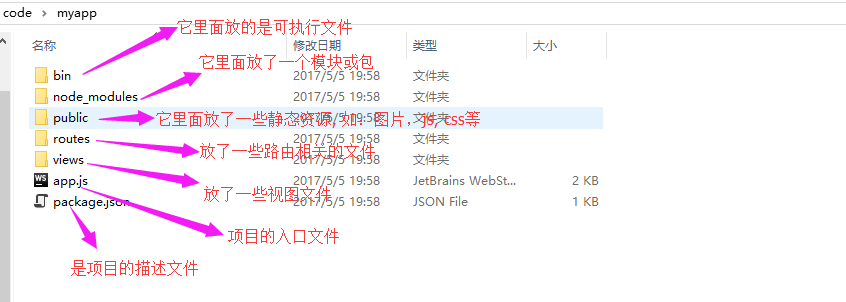
**访问**

访问如下，默认也是3000端口：



**3，项目结构分析**

**整体分析项目结构**



**package.json**

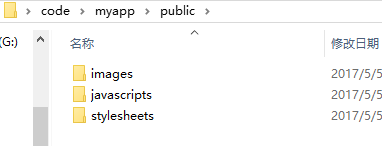


**www文件**



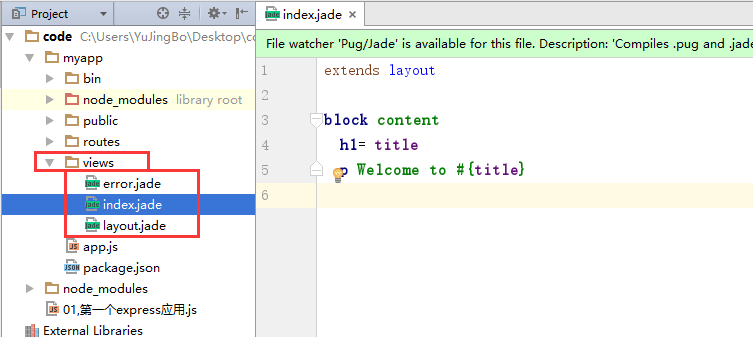
**public目录**

这个目录下面是放静态资源的，整个项目中所用到的图片，css, js都存在在public目录下面。它下面的目录可以修改名字



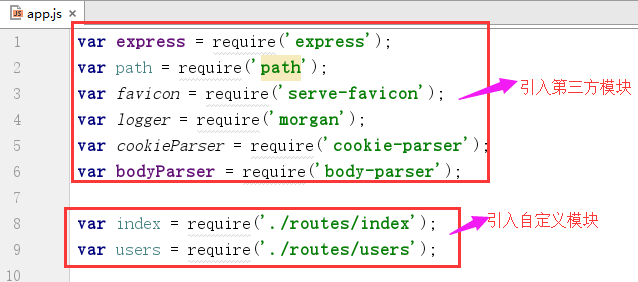
**views目录**

这个目录是用来存在视图相关的文件，在express中，默认是使用jade模板引擎，我们前面说过一个模板引擎ejs



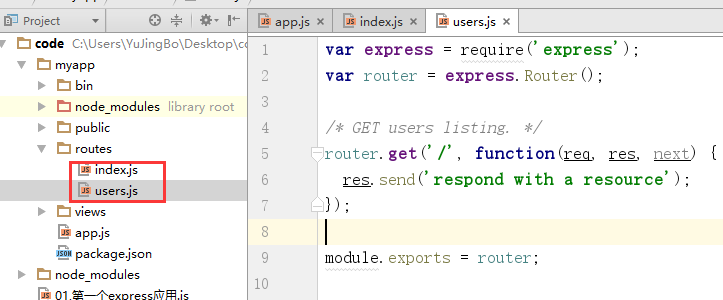
**app.js**

app.js是项目的入口文件



**routes目录**

存在一些路由相关的文件：



**三，路由**

**1，基本路由**

**什么是路由？**

官方解释：路由是指如何定义应用的端点（URIs）以及如何响应客户端的请求。

民间解释：给一个请求，这个请求交给谁来处理，交给的过程就是路由。

**在express中，有两种实现路由的方式：**

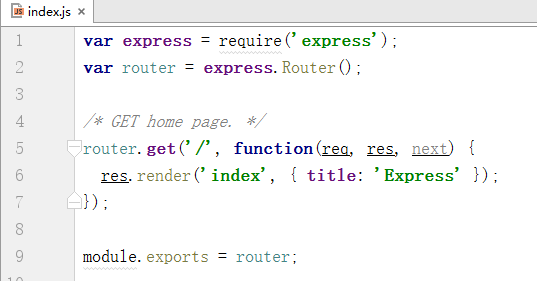
1，针对应用级别的，就是在app对象上设置

2，针对Router实例对象的

**（1）针对应用级别的路由**



**（2）Router实例对象的路由**

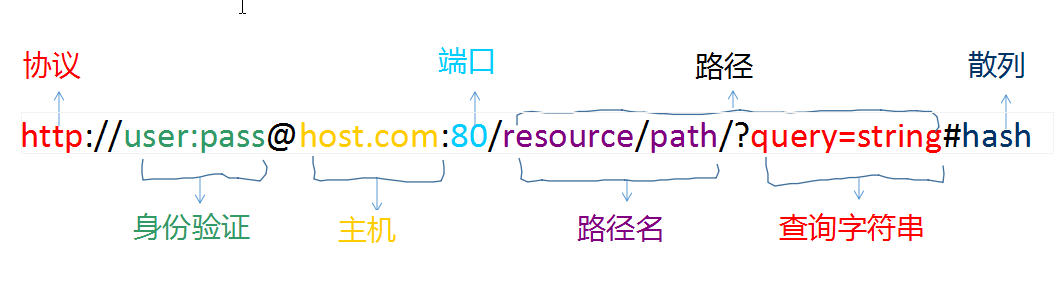


**哪些是get请求，哪些是post请求？**

get: 直接输入一个网址，href，src，表单（method=”get”）

post: 就是表单 通过设置method=”post”

**一个完整的URL如下：**



**总结：**

1，对于上面的两种路由方式，重点掌握第一种。

2，创建一个针对应用级别的路由，是分步骤：

a, 通过express()创建一个app实例

b, METHOD是一个HTTP的请求方法，如get请求或post请求， app.get(), app.post()

c, path是**服务器**上的路径，是url中的路径部分，如 “/” “/user”

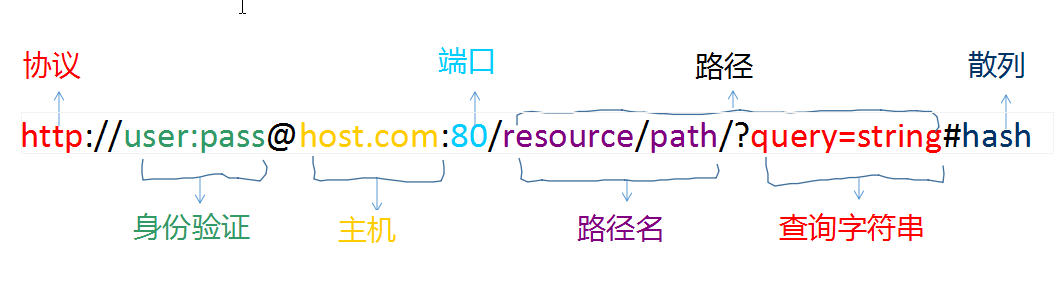
d,callback当路由匹配成功是要执行一个函数，在这个函数中有两个非常重要的参数，req，res, req是指incommingMessage, res是指serverResponse

**2，路由方式**

**什么是路由？**

对于一个请求，交给谁来处理，交给的过程就是路由。

**一个完整的URL？**



路由的方式，不只有get， post还有很多，Express 定义了如下和 HTTP 请求对应的路由方法： **get**, **post**, put, head, delete, options, trace, copy, lock, mkcol, move, purge, propfind, proppatch, unlock, report, mkactivity, checkout, merge, m-search, notify, subscribe, unsubscribe, patch, search, 和 connect。

**一般常见的路由方式有两种：**

get

post

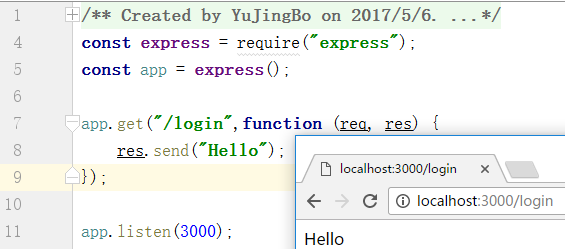
**常见的get请求和post请求的表现？**

get: 直接输入一个网址，href，src，表单（method=”get”）

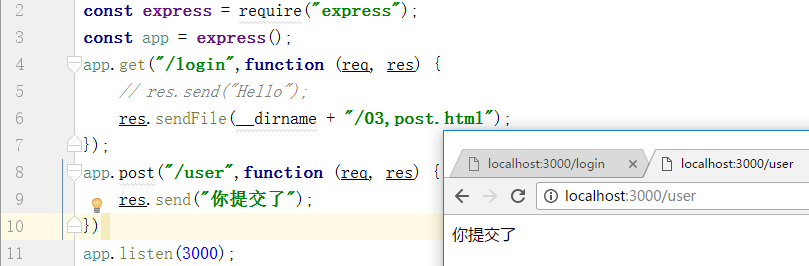
post: 就是表单 通过设置method=”post”

代码演示get和post:

get



post



**3，路由句柄**

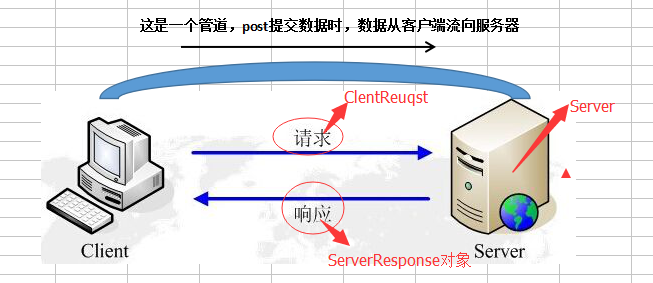
路由句柄就是一个回调函数。如下：

**对于这个路由句柄，它必须要有两个参数，req, res.**

req是指incommingmessage，它表示一些请求信息，这个信息，我们只能获取，不能修改。

res是指serverResponse，它表示响应对象，可以做服务器上的任何事情。

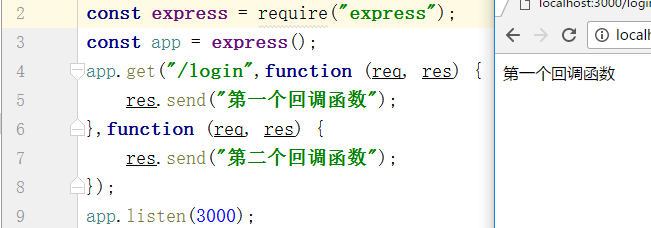
**对于http模块中对应的对象如下：**



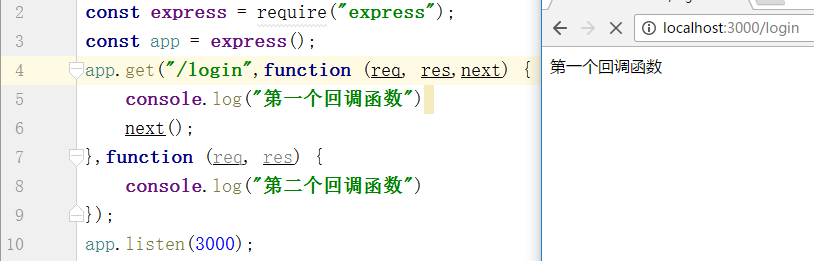
**对于一个路由句柄，可以设置多个吗？**

**可以的**

当我们设置了两个回调函数时，第二个回调函数没有起作用：



想让第二个回调函数起作用，那么就要用到第三个参数，叫next, 需要在第一个回调函数中，调用next()，就会执行下一个回调函数：

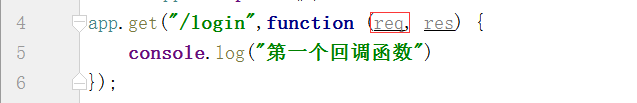


我们还可以在第二个回调函数中，指定next， 这样的话，可以让第三个回调函数起作用。

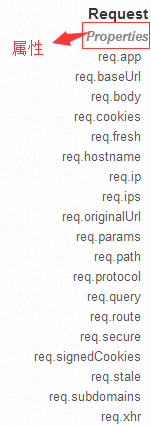
**结论：对于一个路由句柄可以设置多个。**

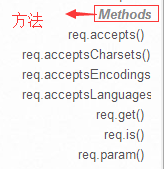
**4，req**

req是作为回调函数的第一个参数



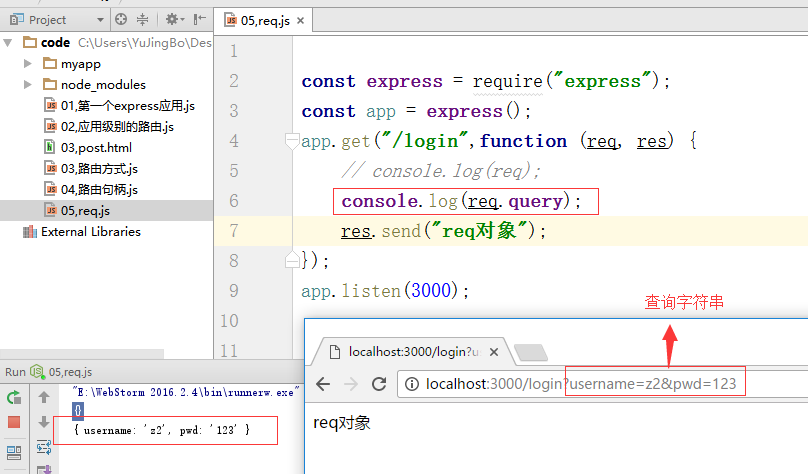
req是incommingMessage对象，它和原生node中req还不太一样，它对原生中的req又进行一次封装，在原来的基础上，又增加了一些属性和方法：





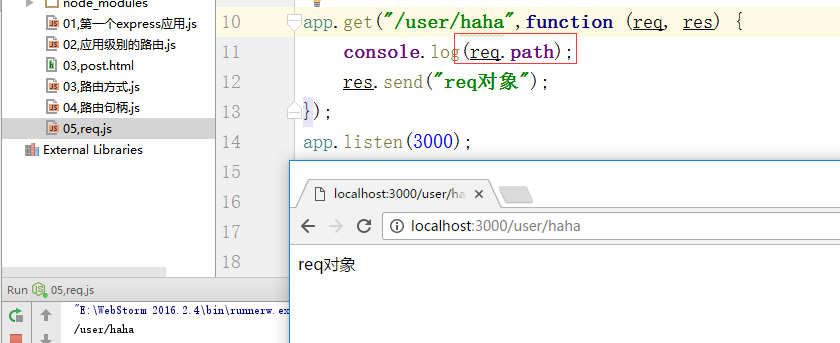
**query**

可以将查询字符串变成对象，如下：



**path**

可以得到路径名：



**5，res**

作为回调函数的第二个参数

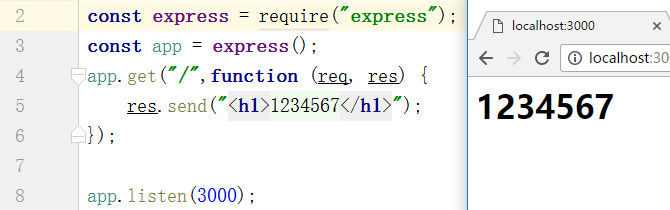
是指ServerRespose对象，响应对象，它和原生node中res还不太一样，它对原生中的res又进行一次封装，在原来的基础上，又增加了一些属性和方法：





**send**

直接发送响应的内容，可以是普通字符串，也可以是html标签，不需要写其它的头信息：



**sendFile**

用来发送文件，我们可以用来发送一个html文件，如下：

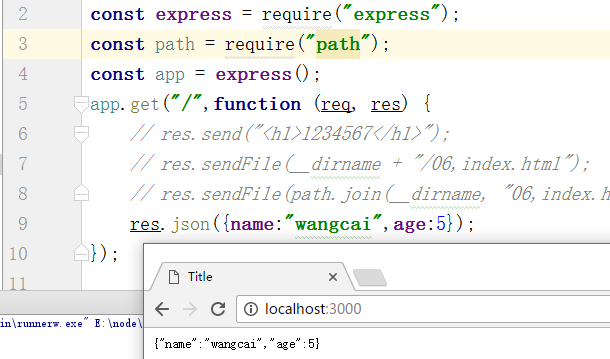


有时候这个/可以会忘记写，可以通过path.join方法来解决：



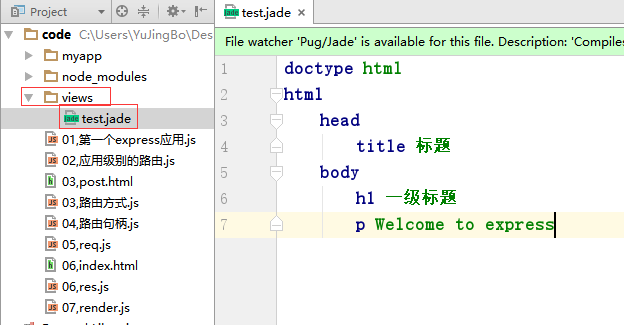
**json**

可以用来发送一个json格式的字符串：

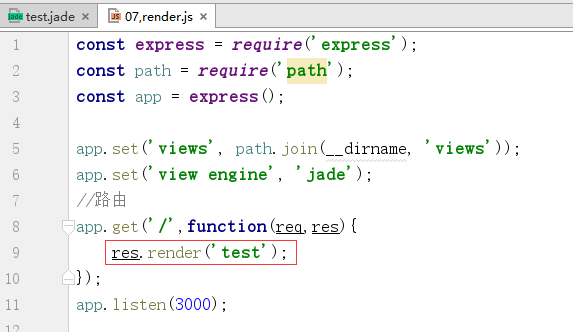


**render**

视图文件如下：



render方法使用如下：

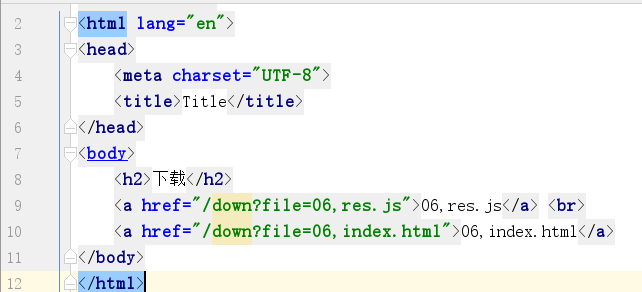


注意的问题：依赖的模块或包都要引进来。

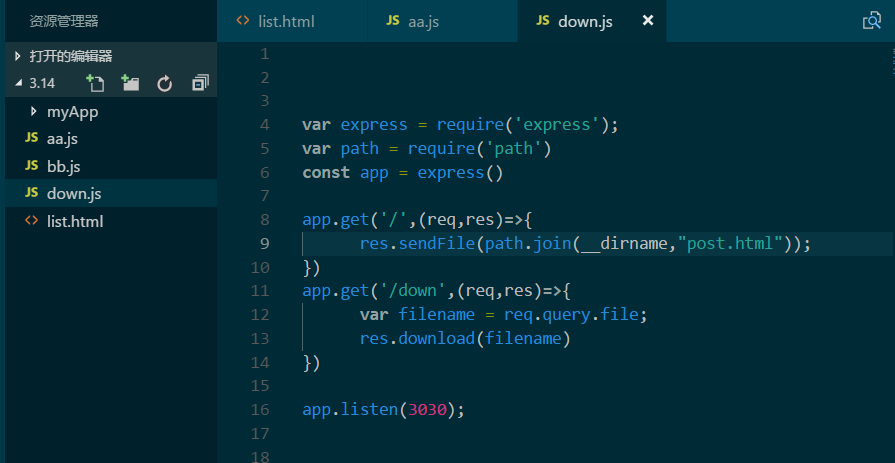
**download**

可以提供下的方法

下载列表页面如下（list.html）：



使用download下载(download.js)：



**redirect**

就是给路由重新定向

重定向方法



**四，中间件**

**1，什么是中间件**

Express 是一个自身功能极简，完全是由路由和中间件构成一个的 web 开发框架：从本质上来说，一个 Express 应用就是在调用各种中间件。

**官方**：中间件（Middleware） 是一个函数，它可以访问请求对象（request object (req)）, 响应对象（response object (res)）, 和 web 应用中处于请求-响应循环流程中的中间件，一般被命名为 next 的变量。

**民间**：所谓的中间件，就是指一个回调函数，在这个回调函数中，有三个参数，req, res, next。它和上午所说的路由句柄差不多。

**中间件可以做什么？**

执行任何代码。

修改请求和响应对象。

终结请求-响应循环。

调用堆栈中的下一个中间件。

**在express中都有哪些中间件？**

应用级中间件

路由级中间件

错误处理中间件

内置中间件

第三方中间件

一个 Express 应用就是在调用各种中间件。路由也属于中间件。

**2，应用级中间件**

所谓的应用级别的，就是指通过app对象来调用。

**如何使用应用级别中间件？**

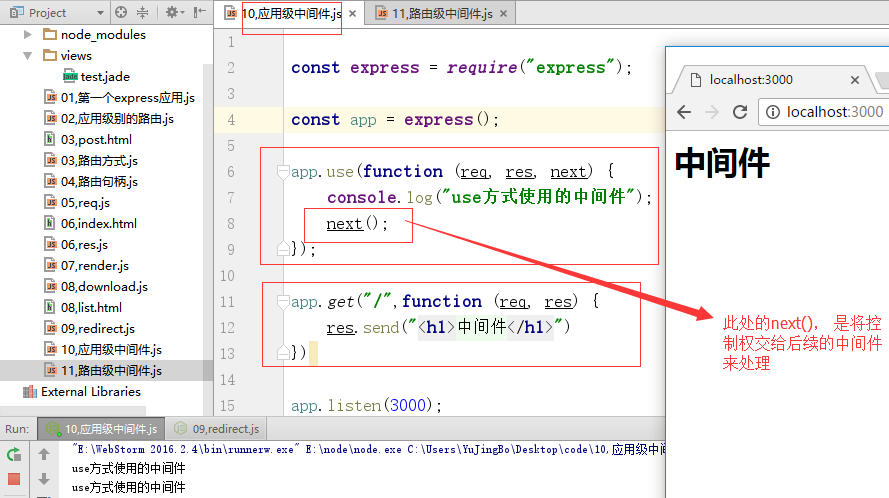
app.use([path]);

app.METHOD(); METHOD是指get, post

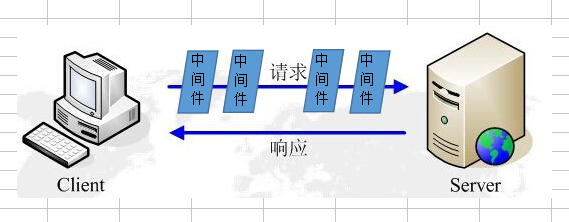
对于app.use()方式的中间件，如果说没有写path，就说明所有的请求都会使用这个中间件。

对于app.METHOD,实际上就是路由，从这个方面来说，路由也是中间件的一种。

USE的用法：



中间件模型：



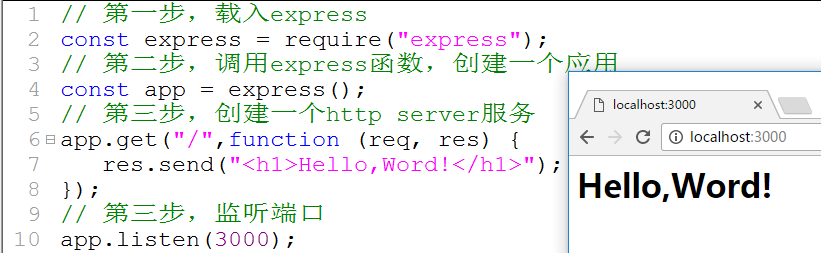
**复习：**

**1，什么是Express?**

它是一个基于node.js的一个框架。Node是一个将JS运行在服务端的一个平台 。

**2，如何使用Express?**

因为Express就是一个模块或包，npm install express安装就可以了。



**3，什么是路由？**

给一个请求，交给谁来处理，那么这个交给的过程就是路由。

**4，路由方式？**

get, post

**5，路由句柄？**

就是一个回调函数，一般要有两个参数，req, res。

req是指IncomingMessage,一些请求相关的信息，这些信息，我们只能获取，不能修改。

res是指ServerResponse，表示响应对应。

**6，req对象和res对象？**

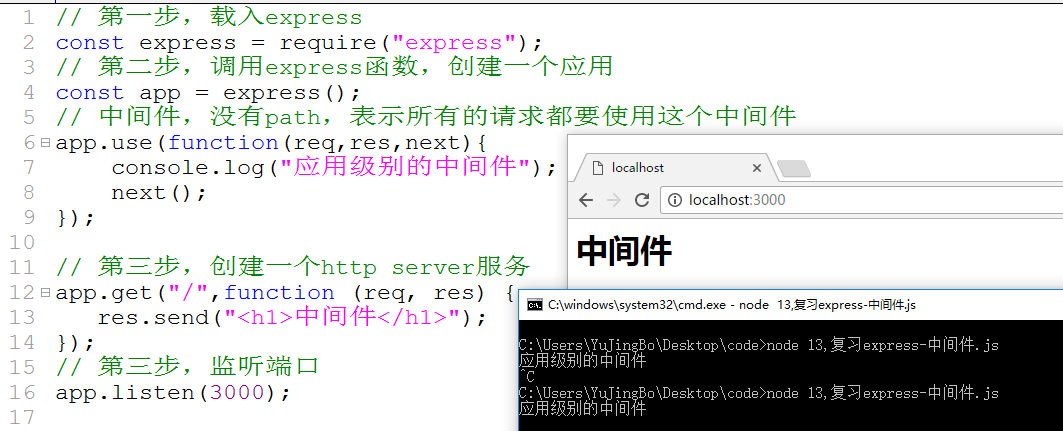
req是指IncomingMessage,一些请求相关的信息。 req.query

res是ServerResponse，表示响应对象。res.send() res.sendFile() res.json() res.render() res.downlaod() res.redirect()

**7，什么是中间件？**

也是一个回调函数，这个回调函数比较特殊，有三个参数，req, res, next。

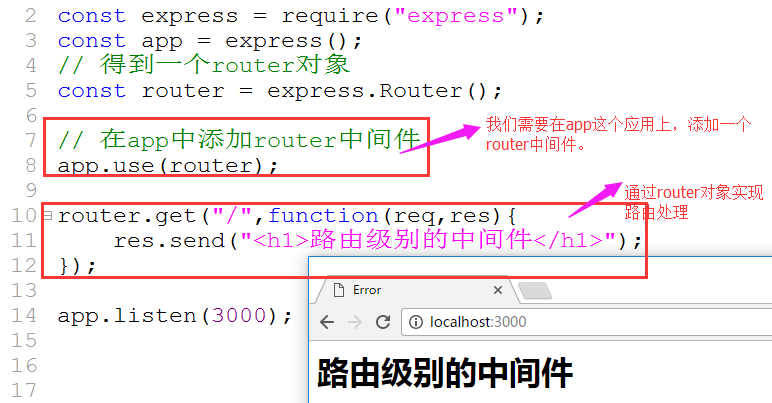
**8，应用级中间件？**



**3，路由级中间件**

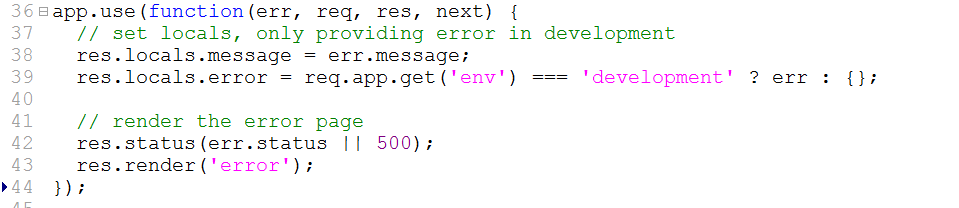
路由级别的是中间件，是指由express.Router对象来调用的。

需要一个Router实例：



**4，错误处理中间件（了解）**

错误处理中间件和其他中间件定义类似，只是要使用 4 个参数，而不是 3 个，其签名如下： (err, req, res, next)。



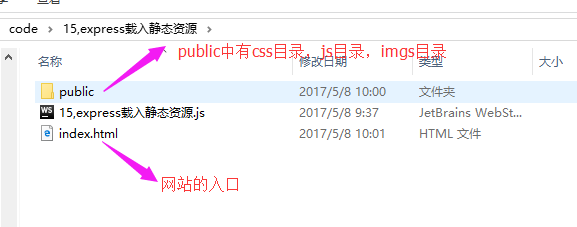
**5，内置中间件**

从 4.x 版本开始，, Express 已经不再依赖 Connect 了。除了 **express.static,** Express 以前内置的中间件现在已经全部单独作为模块安装使用了。

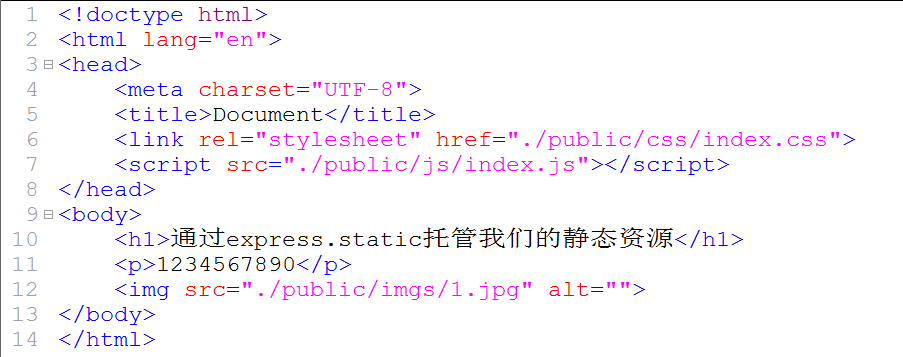
express.static负责托管静态资源的。 说直观一点，它就是用来载入静态资源的。

**具体怎么用？**

静态资源目录如下：

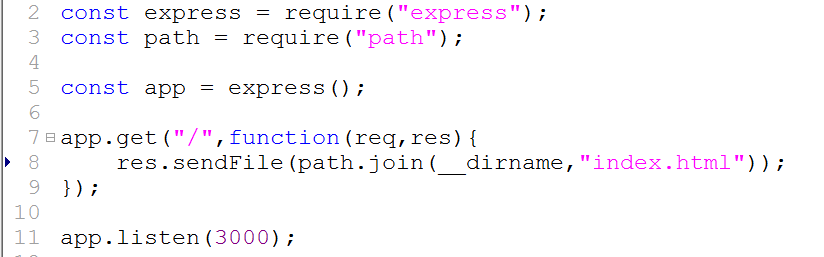


静态资源代码如下：



**还有一个index.css, index.js, 1.jpg**

写代码载入上面的静态文件：

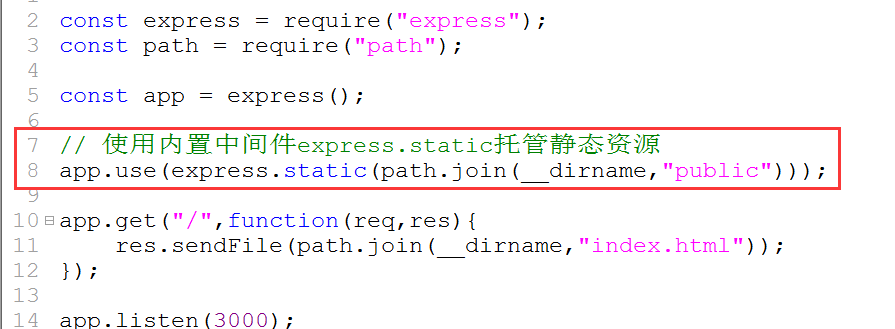


当我们按照上面的写法写时，结果如下：



我们发现，css，img, js没有起作用。怎么办？ 使用express.static解决：

解决如下：



结果如下：



还是不行，为什么，对于图片,css，js它们的路径要相对于public:

（就是说要相对于放静态资源的目录）



最终结果正确，如下：



**6，第三方中间件**

真正开发的时候，使用第三方中间件是最多的。

是别人已经开发好的，被证明是好用的。如body-parser，cookie-parse，express-session。

**使用步骤：**

1，安装

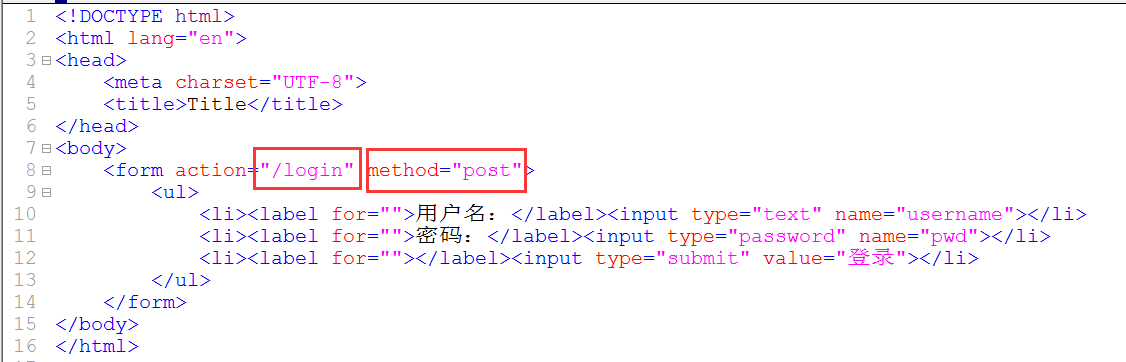
2，引入

3，app.use

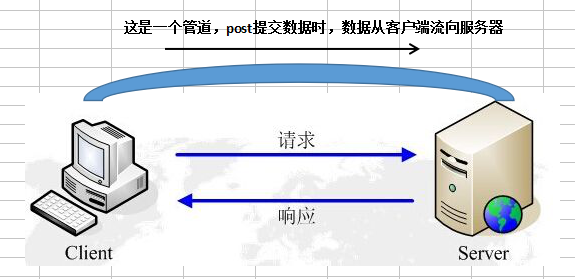
4，调用相关方法

**原生的node获取表单提交的内容**

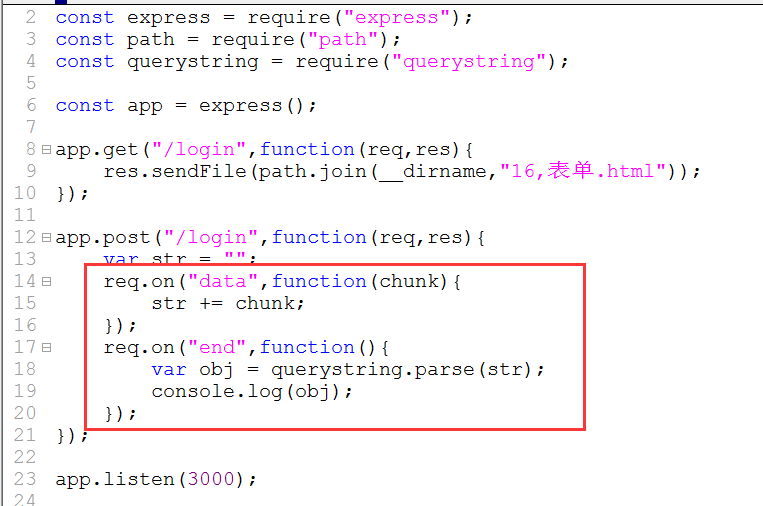
表单代码如下：



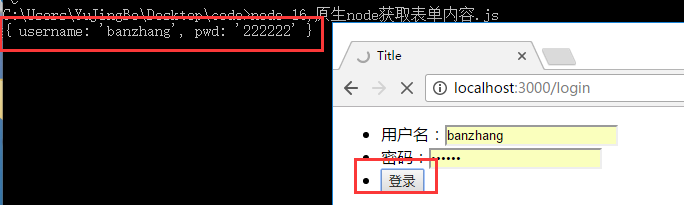
**post提交数据模型：**



**在服务器端获取用户名和密码：**



结果：



**Express获取表单提交的内容**

**第三方中间件：body-parser介绍**

bodyParser中间件用来解析http请求体，解析客户端请求的body中的内容。是express默认使用的中间件之一。

bodyParser.urlencoded则是用来解析我们通常的form表单提交的数据。

**使用步骤：**

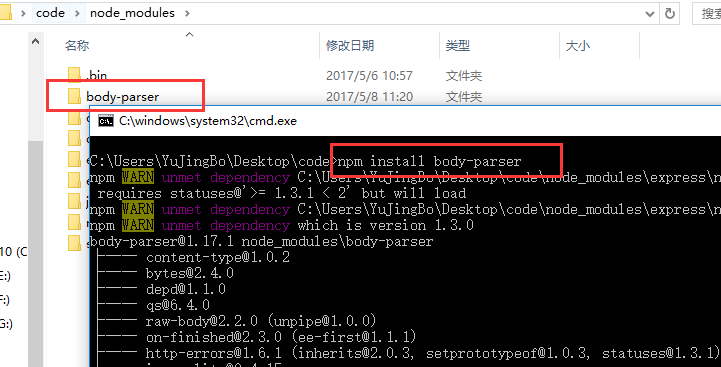
1，安装

2，引入

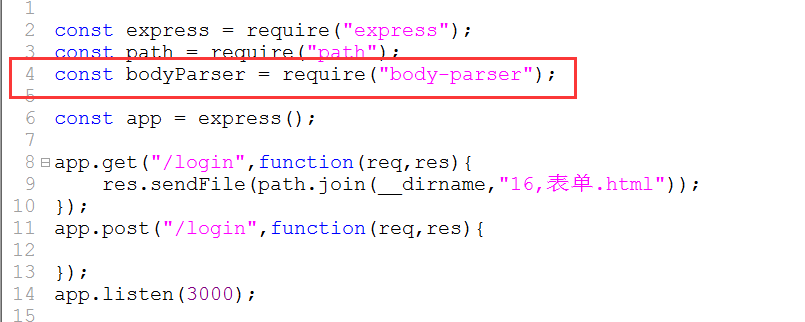
3，app.use

4，调用相关方法

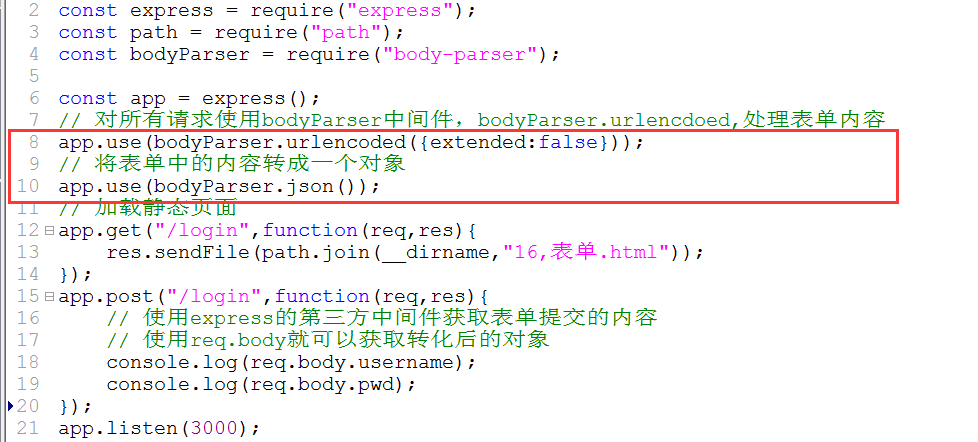
**1，安装**



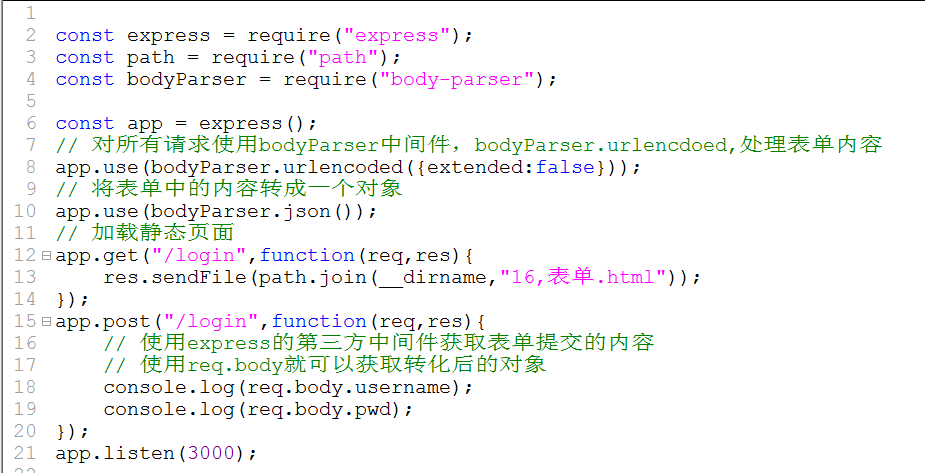
**2，引入**



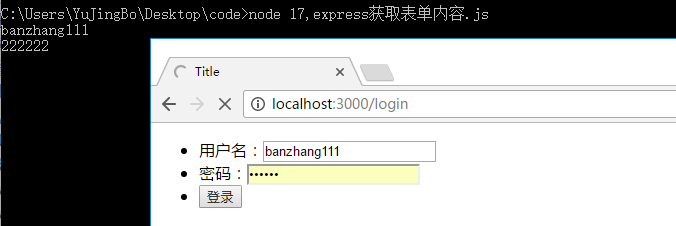
**3，app.use**



**4，完整代码如下：**



结果：



**五，会话机制**

**1，什么是会话**

从不是现实角度来说，打电话就是会话。

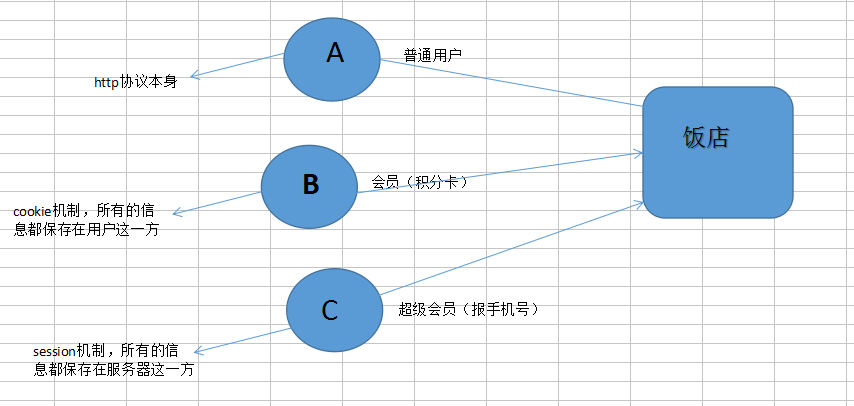
会话机制是Web， 不是express的。

http是一种无状态，无连接的通信协议，没有办法记录状态，这时候，会话机制就出来了。通常有两种形式：

cookie

session

看一个场景：



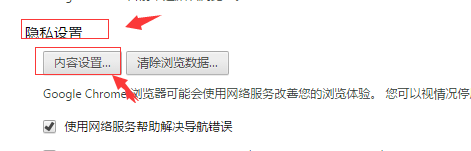
**2，Cookie**

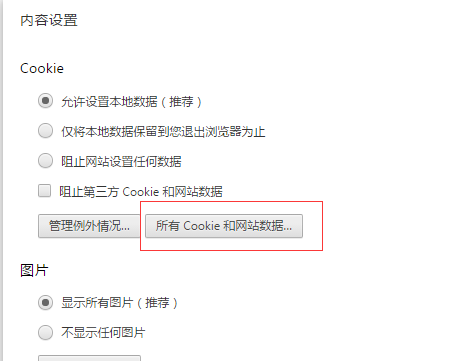
cookie的英文意思：饼干; 小甜点

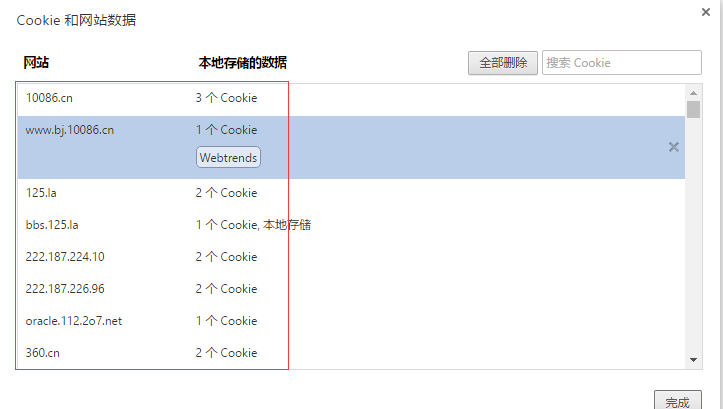
cookie机制是由W3C提出，目前Cookie已经成为标准，所有浏览器都支持。

Chrome浏览器：

进入设置--->显示高级设置









火狐浏览器：

自已找吧

**注意：**

Cookie是基于浏览器的一门技术

不同的浏览器，它们所保存的Cookie不一样

**3，Cookie的基本操作**

**月（1）Cookie保存在什么地方**

**（2）设置Cookie**

**（3）记录用户上次访问时间**

**4，Session**