# 一，初识node.js

## 1，回复一下JS以及DOM

**1，什么是客户端JS？**

前面我们说的JS，只是说了JS的基本的语法。DOM也叫客户端JS。**JS是一门运行在浏览器端的脚本语言。作用：给网页提供行为的。**

**2，客户端JS的运行环境是什么？**

浏览器

**3，客户端JS可以做什么？**

A，写特效

B，写游戏

C，组织页面结构

**4，客户端JS不能做什么？**

A，不能进行文件操作，出于安全考虑

B，不能操作数据库

C，不能与操作系统打交道

.....

客户端JS能力非常弱。

**5，JS只能运行在浏览器端吗？**

不是

Node.js

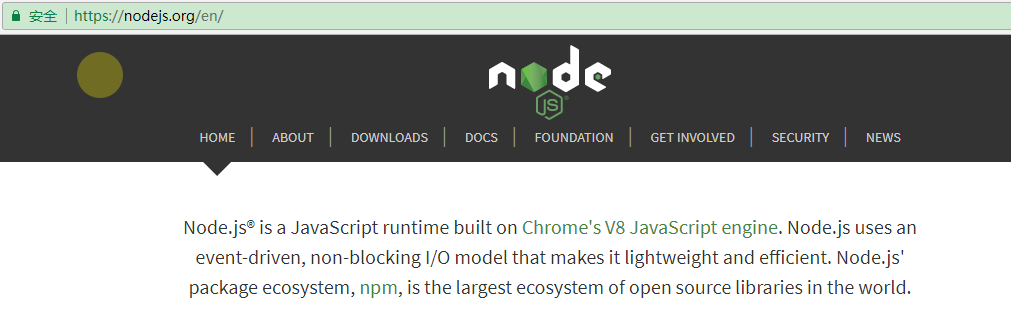
## 2，什么是node.js

Node.js通常有三种叫法：node， nodejs， node.js

官方推荐叫node

官网：<https://nodejs.org/en/>

官网介绍：



**提炼：**

1，nodejs是一个运行环境

2，nodejs是基于v8引擎

3，nodejs使用事件驱动

4，nodejs使用非阻塞I/O模型

5，nodejs有对应的包管理器---npm

中文网：<http://nodejs.cn/>

中文网介绍如下：



**大白话：**

Nodejs其实就是一个JS的运行环境，不是一门编程语言。在nodejs这个运行环境中，可以运行我们写的JS代码。

**对比：**

浏览器也是一个JS的运行环境。Node也是一个运行环境， 它们都是运行JS代码的。

**强调：**

nodejs不是一门语言，只是一个运行环境， 或平台。

## 3，nodejs可以做什么

PHP， JSP， ASP， 传统3P， 是进行服务端编程。Node也是一样的，也是进行服务端编程的。可以做如下：

1，网站/web应用

2，命令行工具

PHP， JSP， ASP， 它们可以做的，NODE基本上都可以做。

## 4，node.js的特点

1，单线程

2，异步式I/O

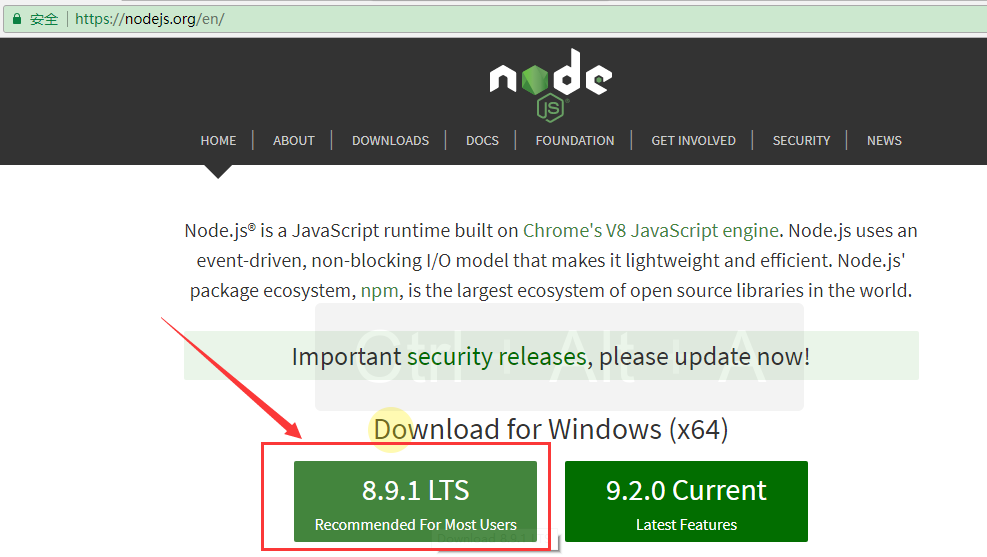
3，事件驱动机制

4，跨平台

## 5，node.js的下载与安装

**下载：**<https://nodejs.org/en/>

**第一步：下载**

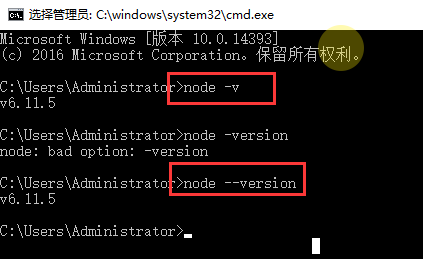


**第二步，下载完后，会得到一个安装包，如下：**



**第三步：双击安装安装包，和安装QQ软件是一样的， 这一步，就不演示了。**

**第四步，测试nodejs有没有安装成功，如下：**

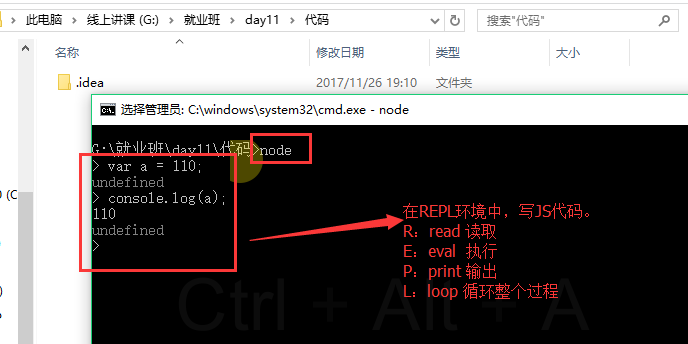


注意。打开一个cmd窗口的几种方式：

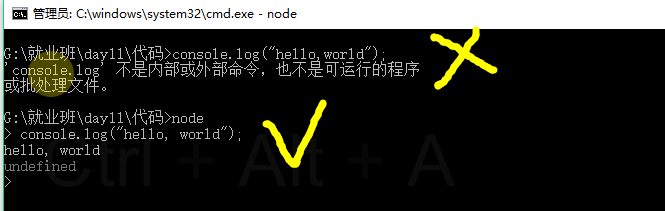
1，按住键盘上的shift键，不松，然后，鼠标右键，最后，选择，在此处打开命令行窗口。

2，按win+r 在输入框中输入cmd， 回车

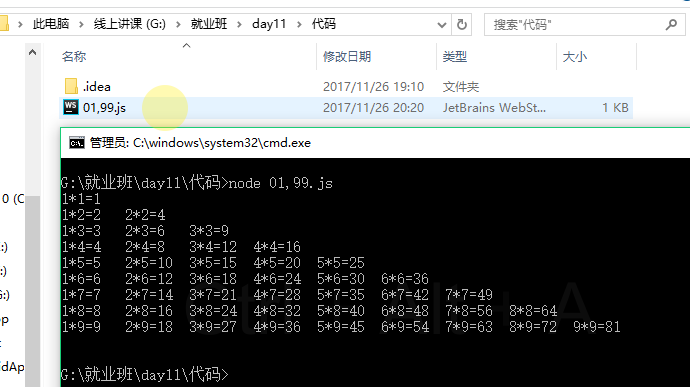
## 6，nodejs初体验



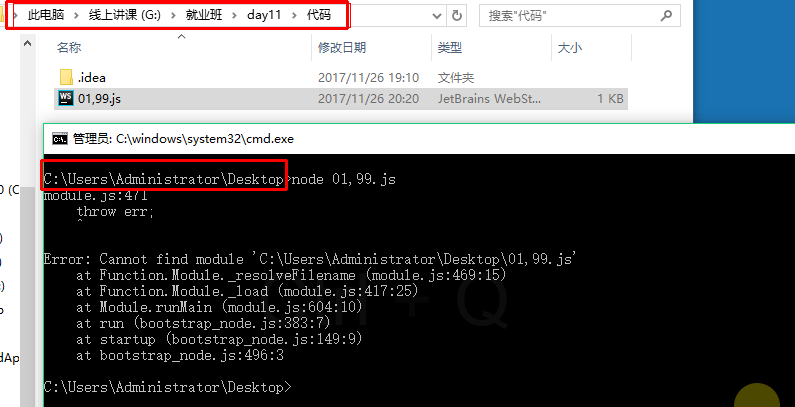
## 7，使用nodejs输出一个hello world!



## 8，使用nodejs输出一个九九乘法表



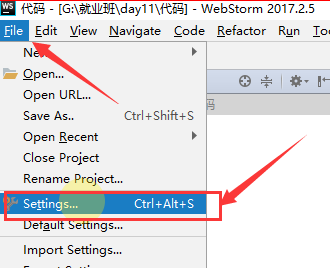
常见问题：

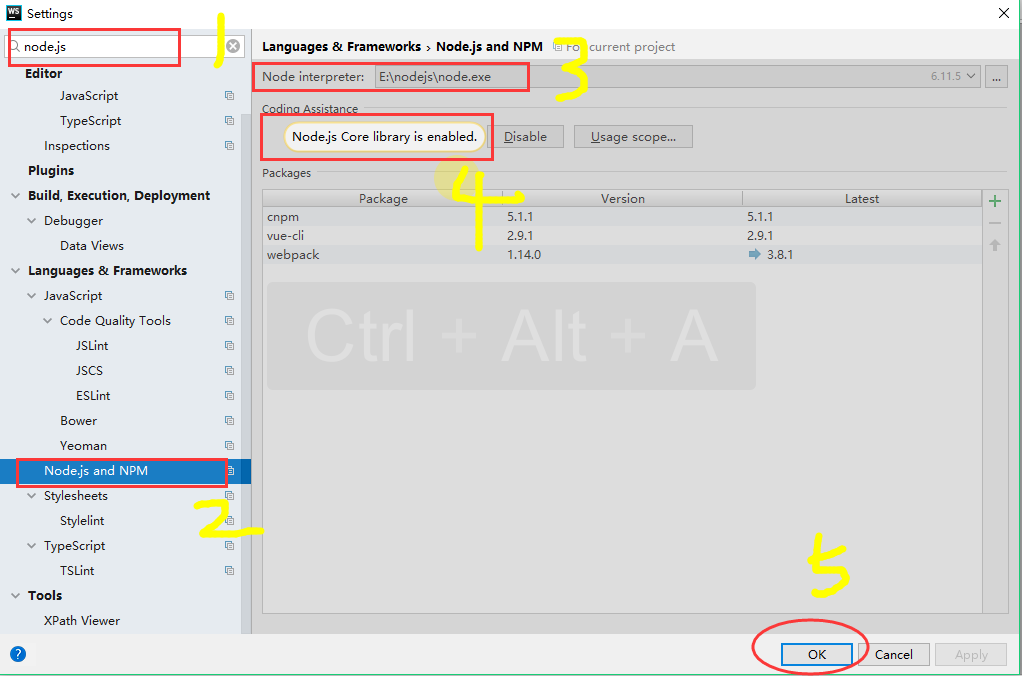


## 9，使用webstrom搭建开发环境

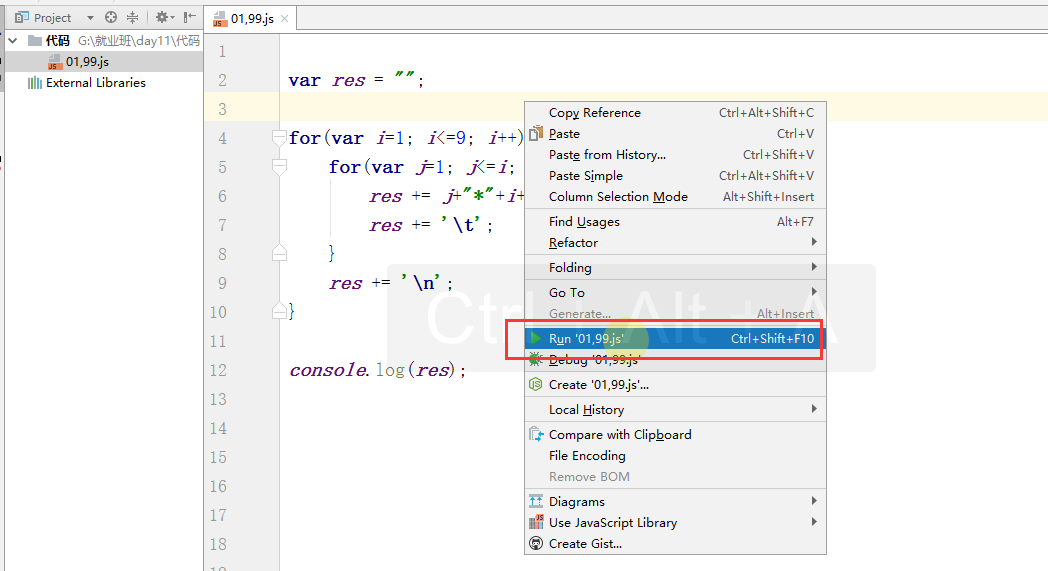
**第一步，安装webstrom。** 不说了。

**第二步，设置：**

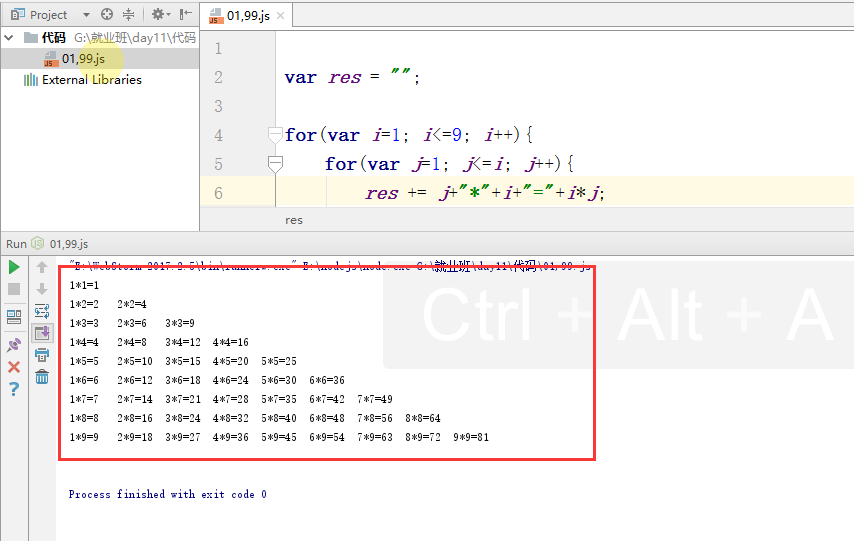




**第三步，写代码，测试，如下：**



结果如下：



# 二，nodejs中的模块机制

## 1，什么模块

从广义上来说，一个文件，就可以叫一个模块。学习过html文件， css文件，js文件，json文件....这些文件，都可以叫模块。模块是nodejs的基本组成部分。

在nodejs中常见的模块有如下三类：

* js文件 后缀是.js
* json文件 后缀是.json
* C/C++文件 后缀是.dll

**为什么要引入模块机制？**

在nodejs中，当实现一个复杂的系统时，不可能将所有的代码都放到一个文件中，为了后期的可重用，可维护。Nodejs引入了模块机制，我们需要哪个模块时，就引入对应的模块， 不需要时，就不引入。这样就达到了可重用，可维护的目的。

## 2，node中常见的模块有哪些

**在node中，模块主要分两类：**

**1，系统模块**

系统提供好的模块，如http模块，fs模块...

**2，用户模块**

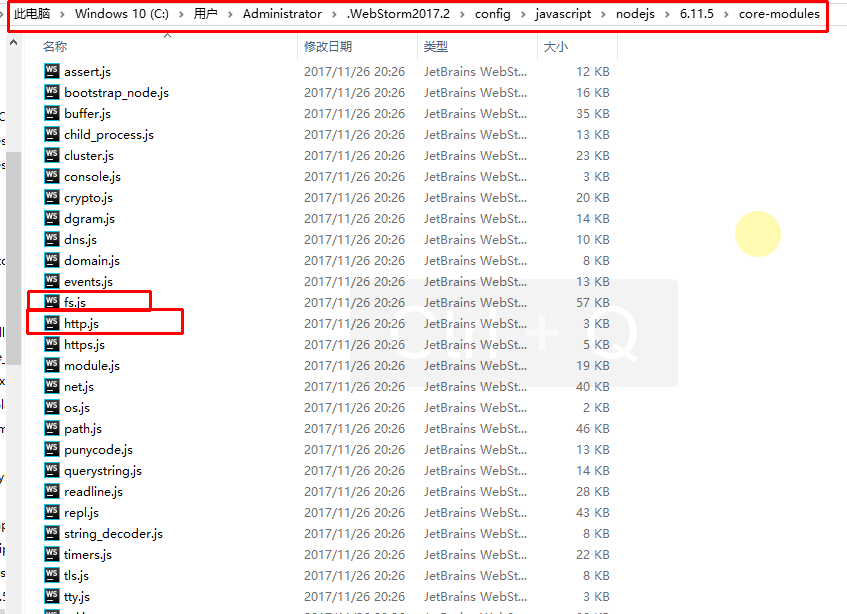
A，第三方模块，别人写好的模块，经过时间的检测，发现很好用，所有很多人都开始使用了，如express

B，自定义模块，需要我们自己去写模块

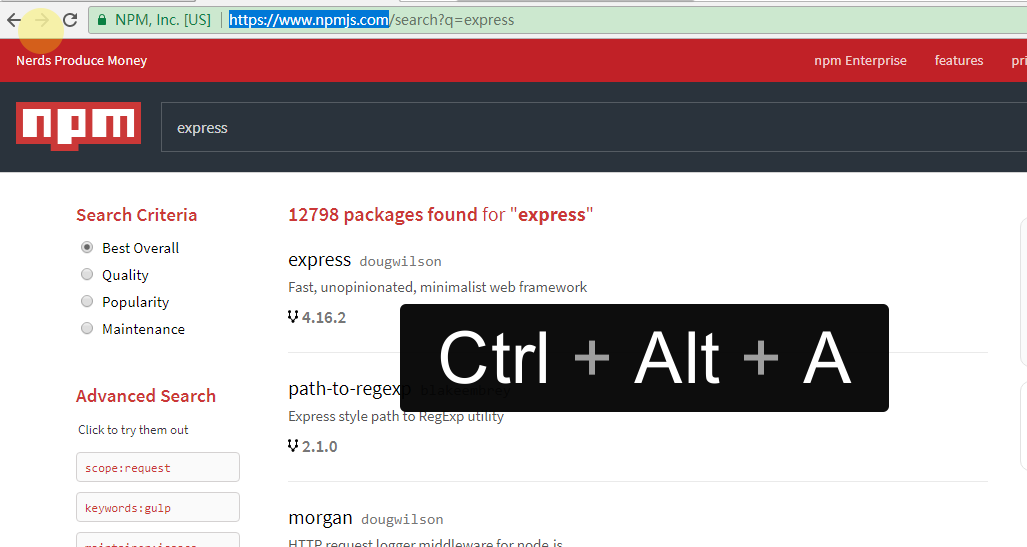
**系统模块和用户模块区别：**

系统模块是编译好的代码，通常以二进制形式存在，引入时，可以直接加入到内存中。

**系统模块安装node时，就安装到了我们电脑上面，位置如下：**



**用户模块：<https://www.npmjs.com>**



## 3，加载模块

require函数，用来引入模块。

### 加载系统模块

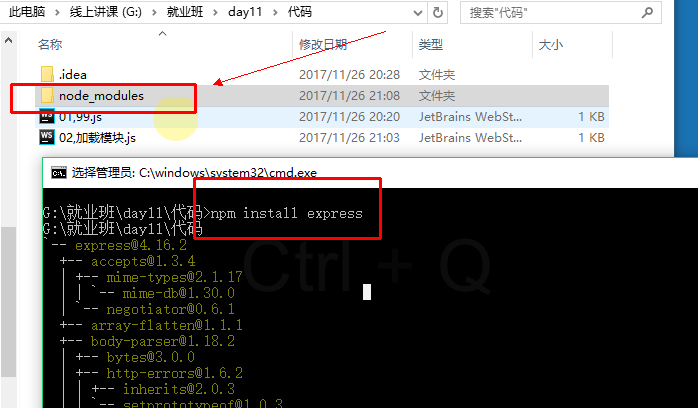


**对于不同的模块，载入时，是有区别的：**

* 系统模块：require(“模块名”)
* 第三方模块：require(“模块名”)
* 自定义模块：require(“./模块名”) require(“./目录/模块名”)

### 加载第三方模块

**第一步：下载第三方模块，如下：**

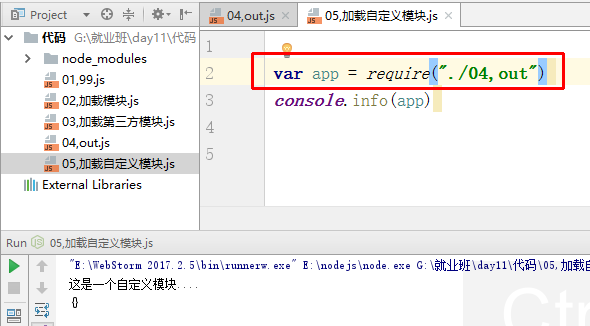


**第二步，在项目中，引入第三方模块**



### 加载自定义模块

一个模块就是一个文件，先自定义一个js文件，如下：



如果当多个不同类型的文件重名时，如：aap.js, app.json。它有一个加载的规则：

目录 ----> js ----> json ----> node 如果一直找不到对应的模块，就报错了。

对于系统模块和第三方模块一般不加后缀，对于自定义模块，也是采用它的加载机制，一般也不写后缀。

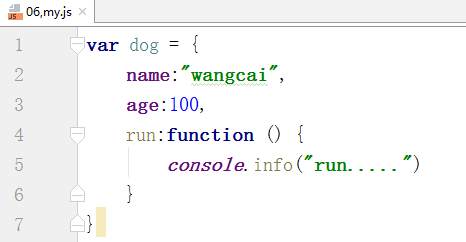
## 4，自定义模块

对于系统模块以及第三方模块，直接拿来使用就OK了，重点是自定义模块。

**自定义模块的使用流程：**

创建模块---->导出模块---->引入模块---->使用模块

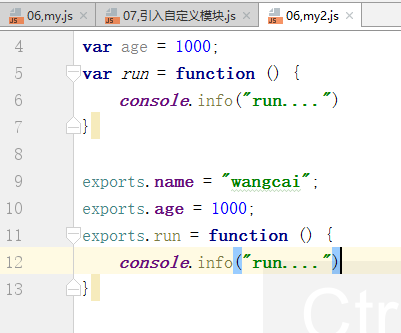
**创建模块，如下：**



**导出模块，如下：**



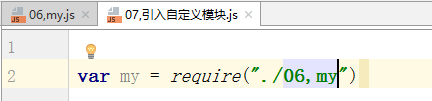
除了moudel.exports，还有一种方式用来导出模块：exports



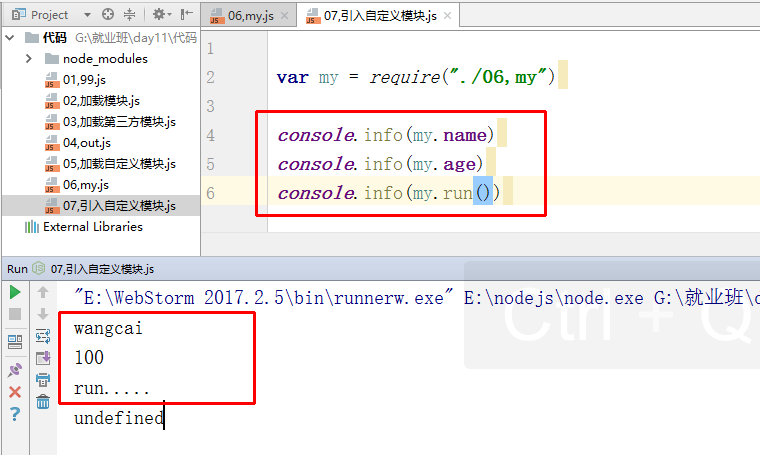
**区别：**

当导出一个对象时，使用module.exports， 当导出一个变量或函数时，使用exports

**引入模块，如下：**



**使用模块，如下：**



# 三，文件系统

在Nodejs中，其实就是一个模块。这个模块叫fs模块。

node中有两个核心模块：

* fs模块
* http模块

## 1，什么是文件系统

filesystem fs 是Node中的一个系统模块

通常所说的文件系统，主要就是对文件进行操作。CRUD操作。

C: create 增

R: read 读

U: update 改

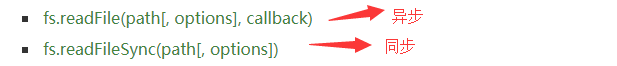
D: delete 删

## 2，常见的文件操作

在中文网中找文档，在文档中，找到文件系统，如下：



### 读取文件



对于异步，是通过回调来实现的，对于同步，是没有回调的。



注意问题：如果我们没有指定第二个参数，那么最终我们得到的是一个buffer。

对于异步操作，是通过回调函数实现的，第一个参数是errr， 如果出错了，错误信息都保存在err这个参数中。

出错情况，如下：

同步，如下：



小结：

1，对于所有的文件操作，都有两个版本。

2，对于同步的，返回值就是文件的内容，对于异步的，文件的内容是放在回调函数中的第二个参数中。

**关于同步和异步的执行顺序问题：**

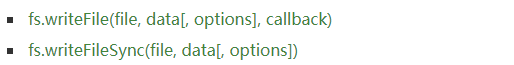
对于同步的执行顺序：



对于异步的执行顺序：



### （2）写文件操作

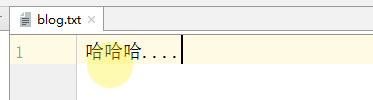


**A，覆盖式的写文件**

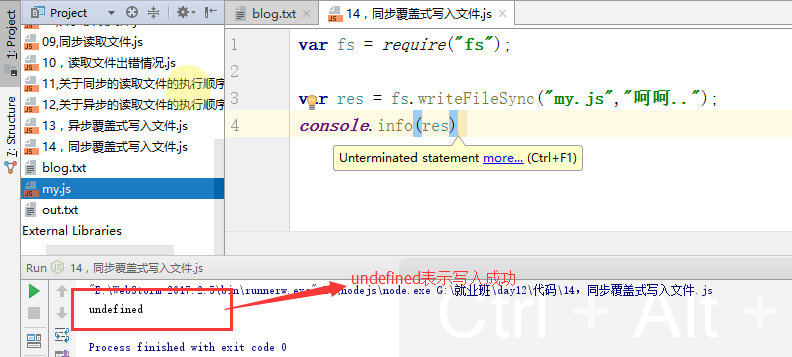
**异步操作：**



结果如下：



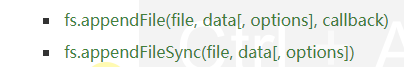
**同步操作，如下：**



注意：undefined表示写入OK。

**B，追加式的写文件**

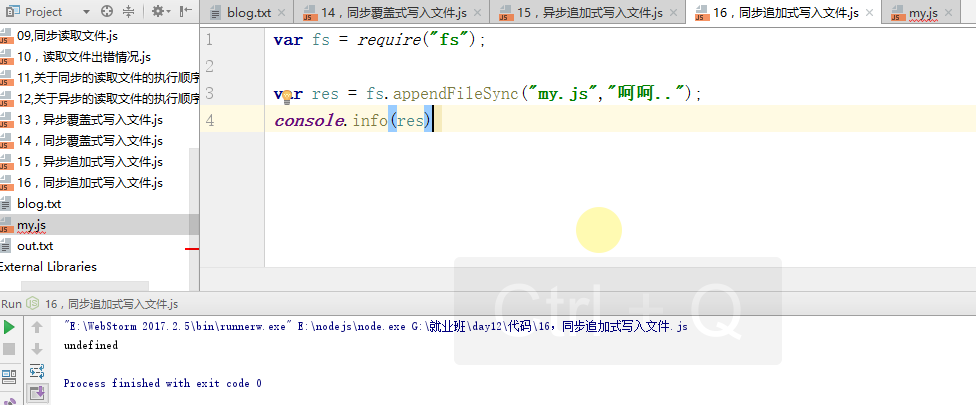
**Append**



**异步操作：**

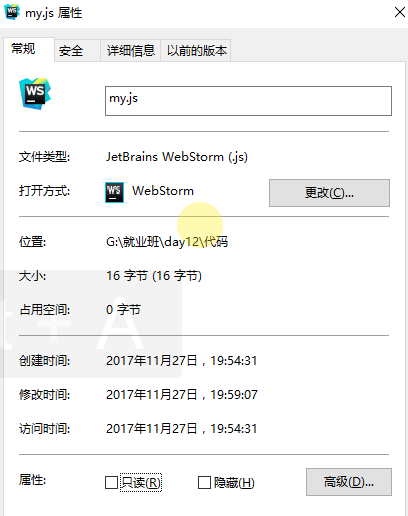


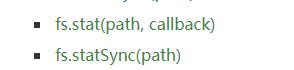
**同步操作，如下：**



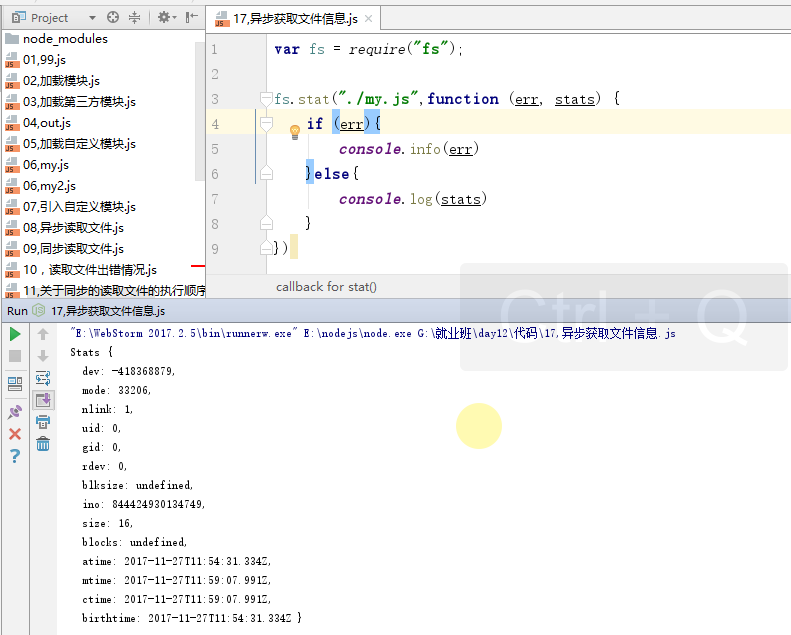
### 获取文件信息

一个文件，都有一些信息，如下：

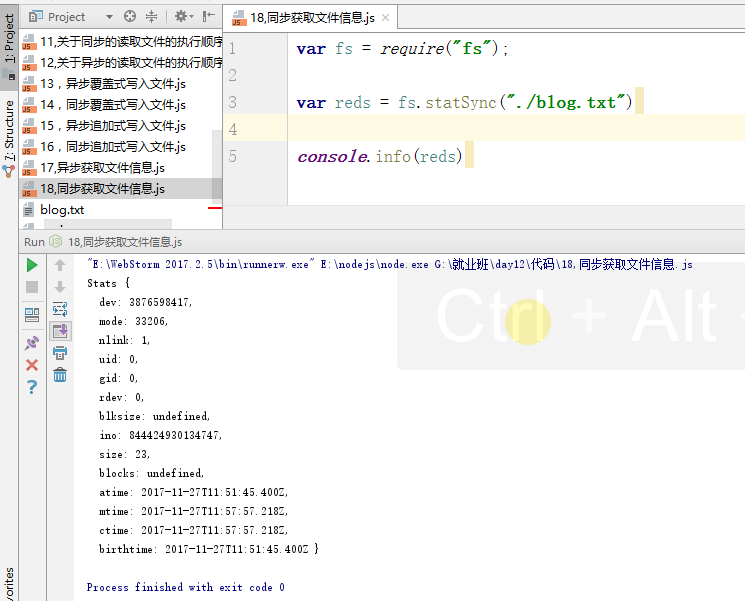




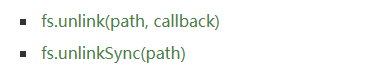
异步获取文件信息，如下：



同步获取文件信息，如下：

‘

### 删除文件



## 3，目录操作

对于目录操作。CRUD操作。

C: create 增

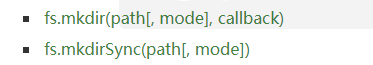
R: read 读

U: update 改

D: delete 删

### 创建目录

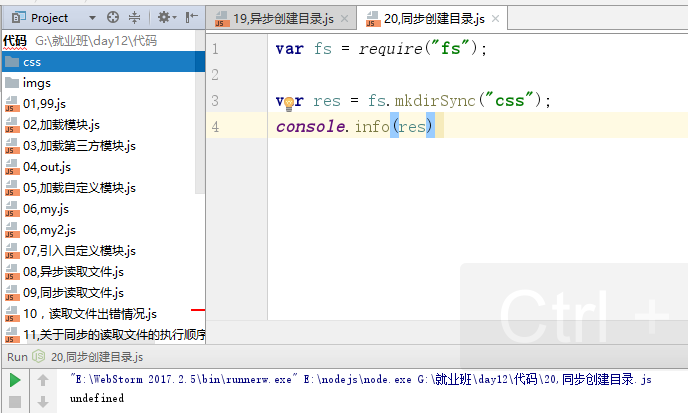
Mkdir



**异步操作：**



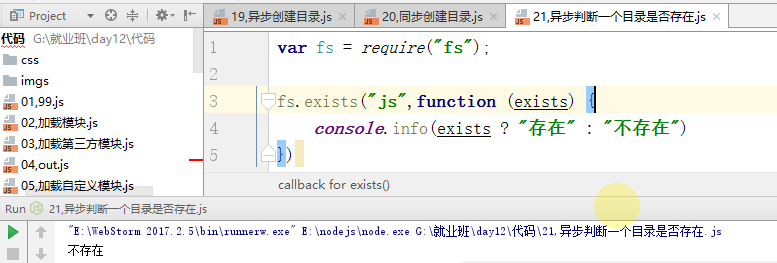
**同步操作：**



### （2）判断一个目录是否存在



异步操作，如下：



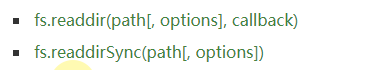
同步操作，略。

### 删除目录



### 读取目录

read dir



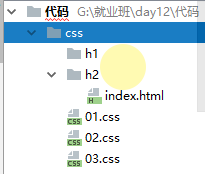
异步操作：



同步操作：略。

### 递归删除目录

对于下面的一个目录结构，如何删除？



分析：

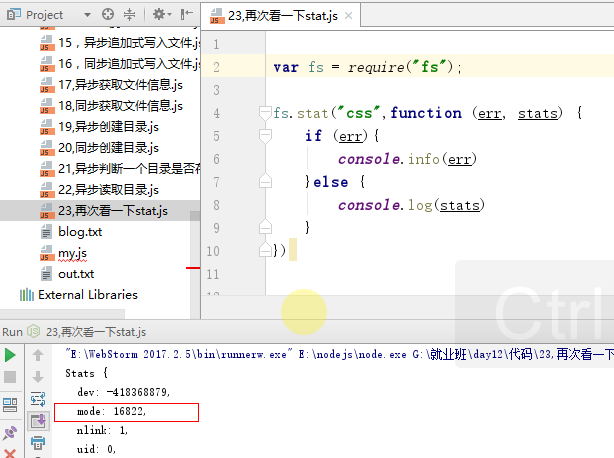
1，删除目录时，用rmdir， 删除文件时，用unlink

2，判断是一个文件还是一个目录

3，如何是一个文件：stat

再看一个stat:



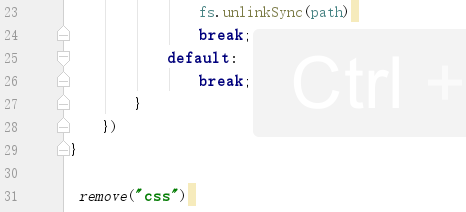


对于目录和文件，它的mode值是不一样的，可以去区分出如一个是文件，哪一个是目录。

封装一个函数，功能就是递归删除一个目录。

代码如下：





# 四，使用node.js进行web应用开发

## 1，什么是HTTP协议

**客户端：**

APP

电脑

**浏览器**

**服务器：**

服务器也是一台电脑，非常昂贵，性能也非常好，它上面的装的操作系统我和们用的电脑上的操作系统也不一样，一般是linux。并且它上面装的应用软件和我们的也不一样。Appche，就可以对外提供web服务子，装了Mysql的软件，就可以对外提供数据服务了....

**我们的电脑能当作一个服务器吗？**

答：肯定可以的

**H**yper**T**ext **T**ansfer **P**rotocol 超文本传输协议。

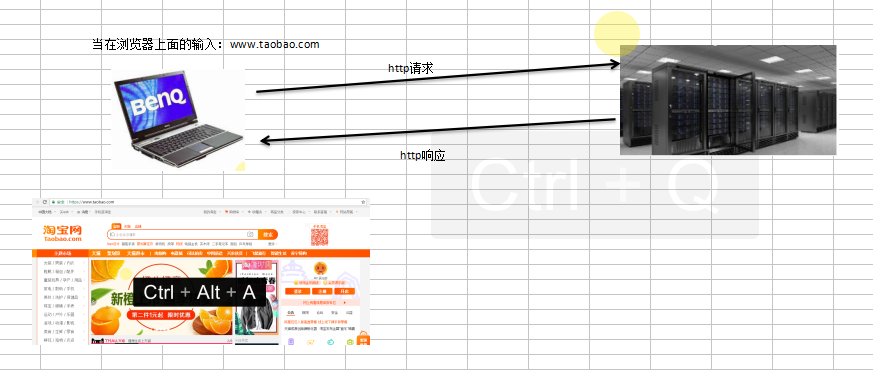
人--人 交流 ： 汉语， 英语 规则 协议

人--计算机 交流 ： JS JAVA PHP 规则 协议

计算机与计算机 交流 ： HTTP 规则 协议

Http协议客户端与服务器进行交流的协议。http协议包含两个部分：http请求，http响应。



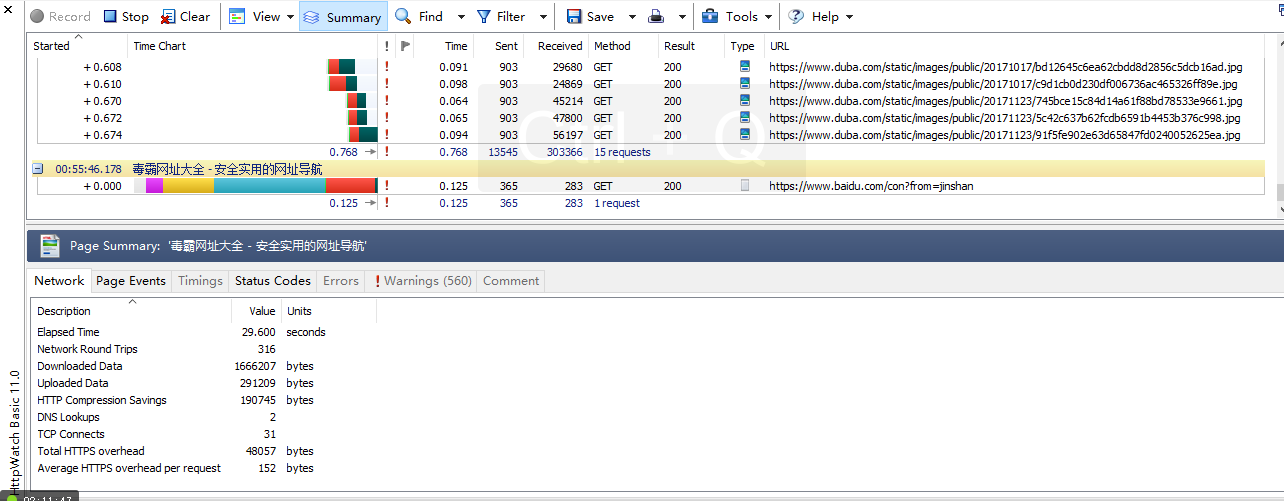


这也是上网的本质。

如果查看http请求与响应的细节过程：

1，浏览器自带的一个工具

2，专业点的工具，叫httpwatch， 目前只支持火狐浏览器与IE浏览器。火狐浏览器只支持低版本的浏览器。这个工具，直接百度下载，傻瓜式安装就OK了。



## 2，wampServer软件介绍

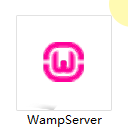
因为，node里面没有web容器概念。上面这个软件是开发PHP用的。

* W: window
* A: apache
* M： mysql
* P: PHP

对于上面的几种服务器软件，如果单独装的话，配置麻烦，所以有人就把上面的三种软件集成到一个软件包中，只需要下载这个软件包就OK了。



安装完后，在桌面上有如下图标：



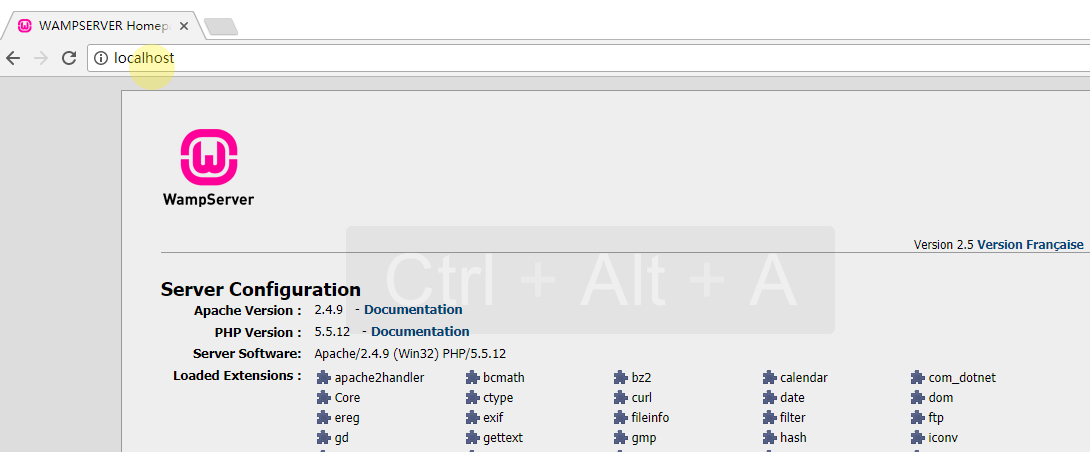
安装完后，双击运行，出现如下标志，表示OK：



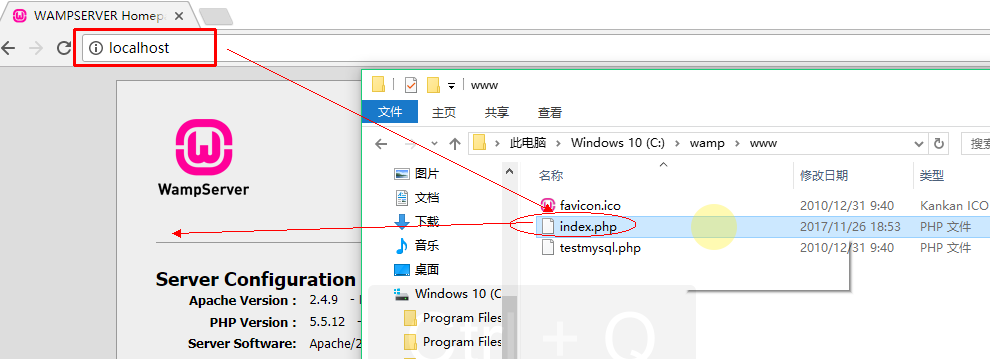
此时，我们的电脑就是一台服务器。我们的电脑又充当了客户端，又充当了服务器。

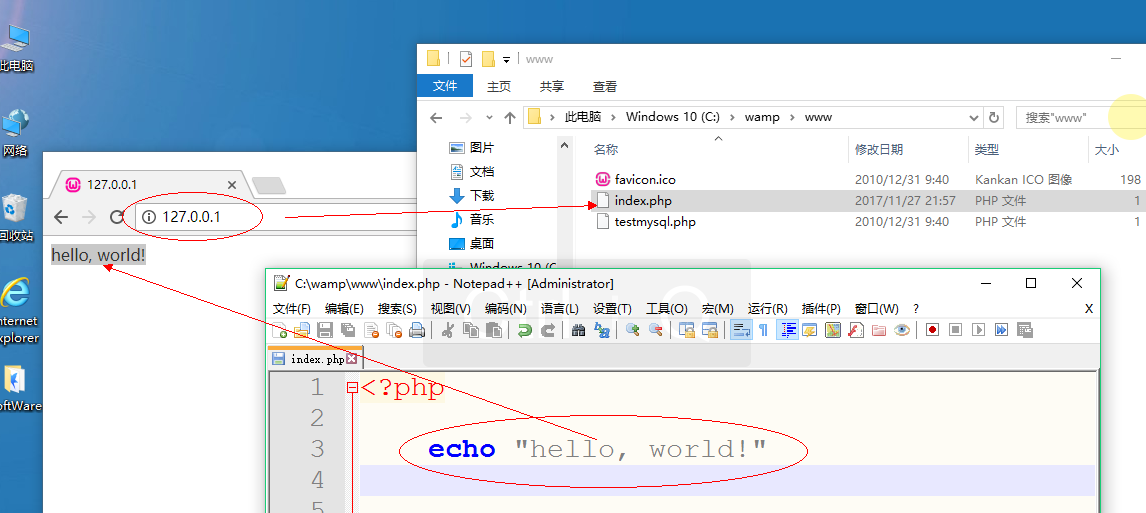
在自已的服务器，IP：127.0.0.1 也有对应的一个域名：localhost

现在，打开浏览器，在浏览器中输入loclhost，127.0.0.1，就表示去访问本机服务器：

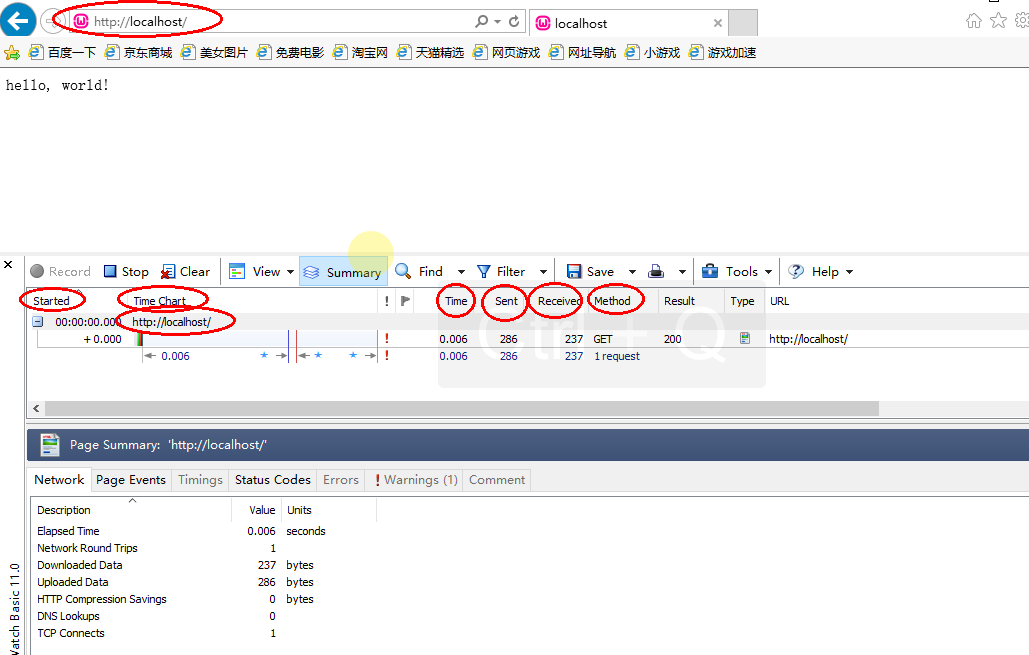


上面我们所看到的就是服务器，给我们返回的一个页面，浏览器拿到空上页面后，进行渲染，进行显示。





Httpwatch 工具介绍：



在一个http请求过程中包含两个部分：http请求，http响应。

到目前为止，htttp协议分HTTP请求和HTTP响应。

模型图：



HTTP请求

1. 分：

请求行

请求头（消息报头）

请求正文

HTTP响应

三部分：

响应行（状态行）

响应头（消息报头）

响应正文

关于HTTP协议的细节内容，后面找个时间专门说说一下。

## 3，HTTP请求

1. 分：

请求行

请求头（消息报头）

请求正文

## 4，HTTP响应

三部分：

响应行（状态行）

响应头（消息报头）

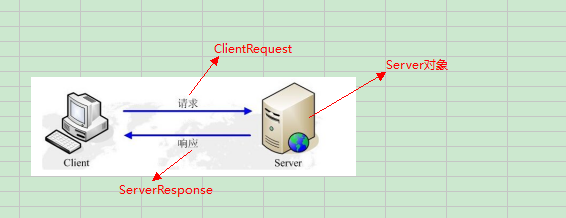
响应正文

## 5，Server对象



在node.js中，HTTP模型都地通过对象来实现的。Server就一个对象，请求也是一个对象，响应也是一个对象。

node中http模型中对象模型图，如下：

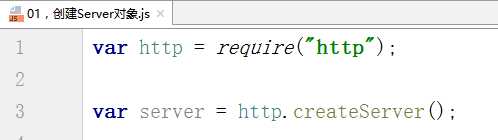


接着，我们学习一下，Server对象。

**第一步，如何创建出一个Server对象？**

答：http.createServer() 可以创建出一个Server对象

代码如下：



**第二步，注册用户请求事件**



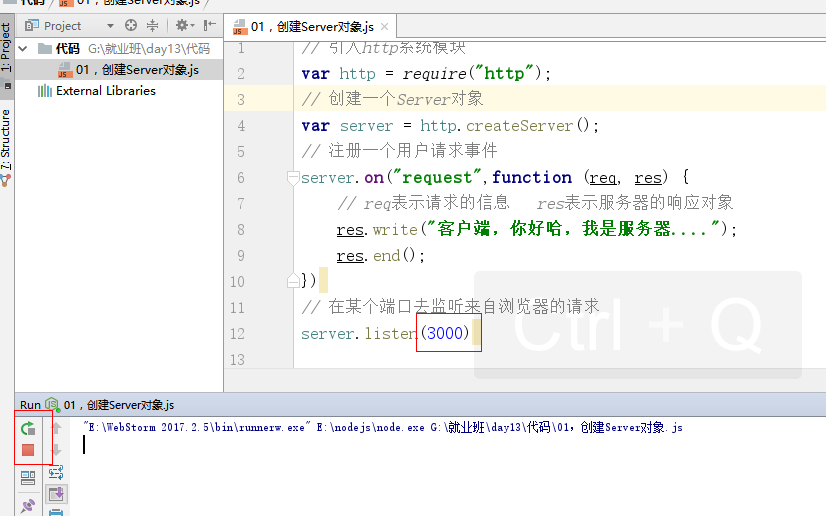
**第三步，在某个端口监听来自浏览器的请求**



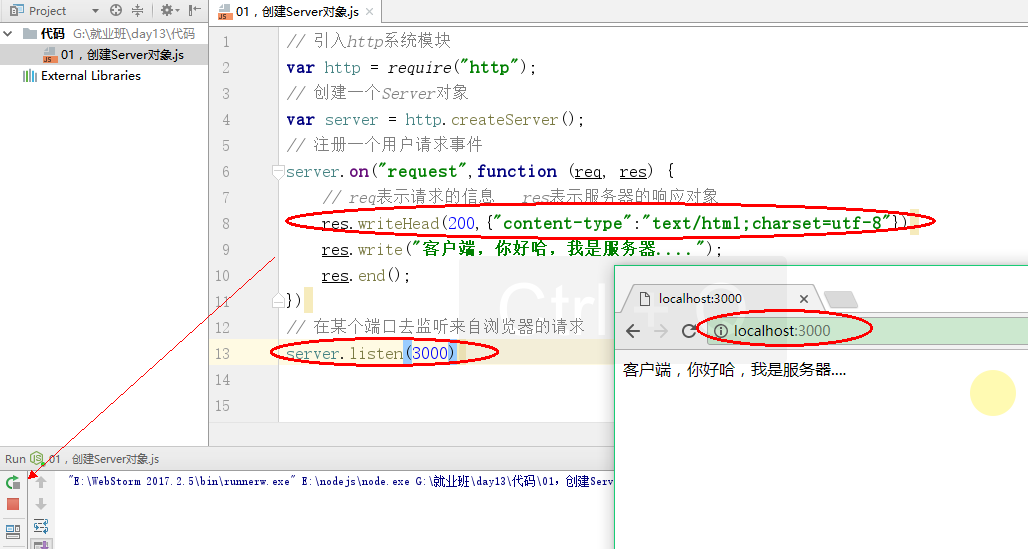
完整代码如下：



接着，运行代码，说明，服务器已经开启了：



上面已经把服务器开启了，接着，就要去请求服务器，在浏览器中请求，本机的服务器的IP是127.0.0.1，这个IP它对应的一个域名：localhost



注意：

当你修改了服务器的代码时，你需要去重启服务器，否则没有效果。

对于上面创建一个服务器的写法，之所以那样写，是为了让你更好理解过程，一般情况下，我们都会使用它的简写方式，如下：

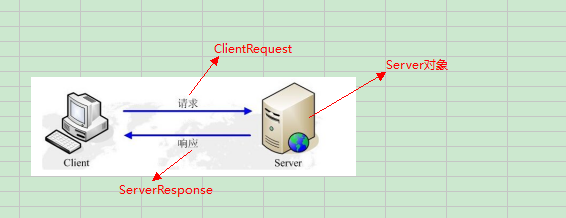


官方的写法：



## 6，ServerResponse对象

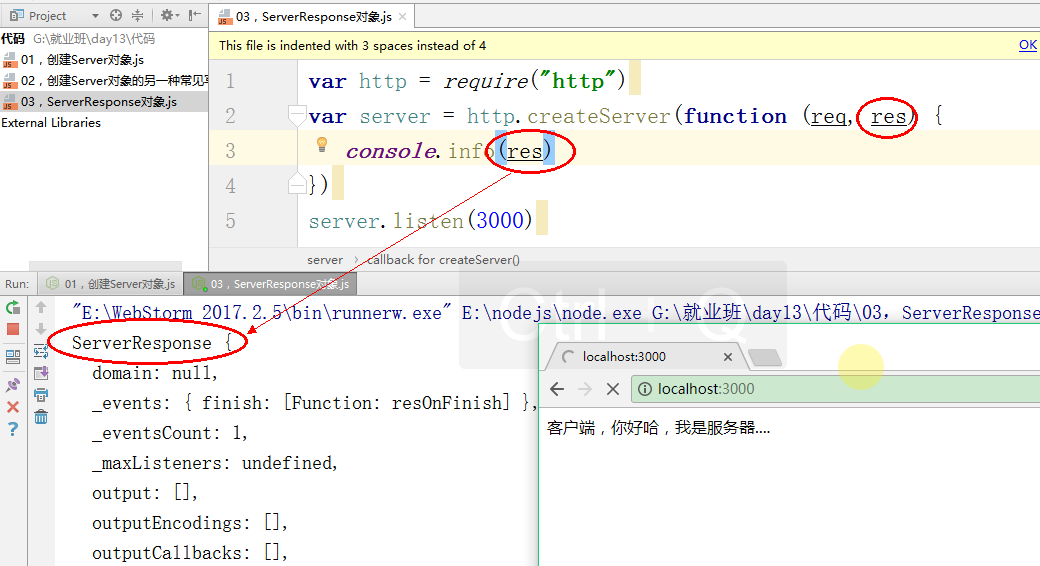
再看一次，http模型图：



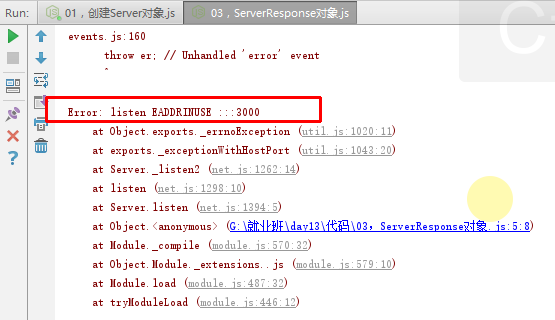
**第一步，如何创建出ServerResponse对象？**

答：不用手动创建，是自动创建的，创建完毕后，它是作为监听器的第二个参数存在的。

代码如下：



**注意，当你看到如下错误时，说明你开启了两个服务器，并且这两个服务器监听了同一个端口，如下：**



**对于ServerResponse对象有哪些方法与属性呢？**

1,writeHead 写入响应头

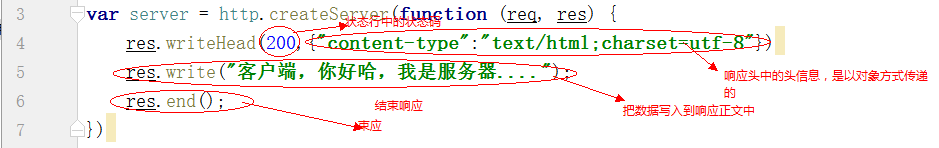
2,write 把数据写入到响应正文

3,end 完成响应，同时，也可以把数据写在end中

4,setHeader 设置特定的http头信息

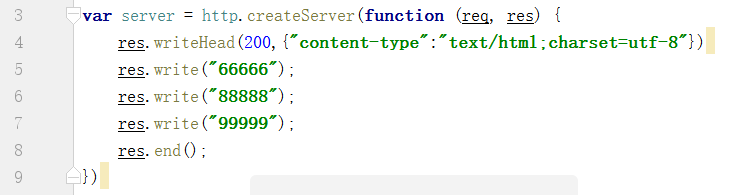
5,getHeader 获取指定的http头信息

代码如下：

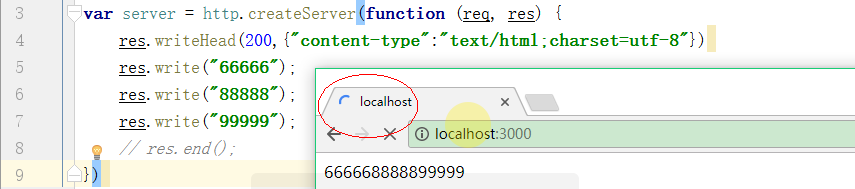


对于ServerResponse这个对象需要注意的问题，如下：

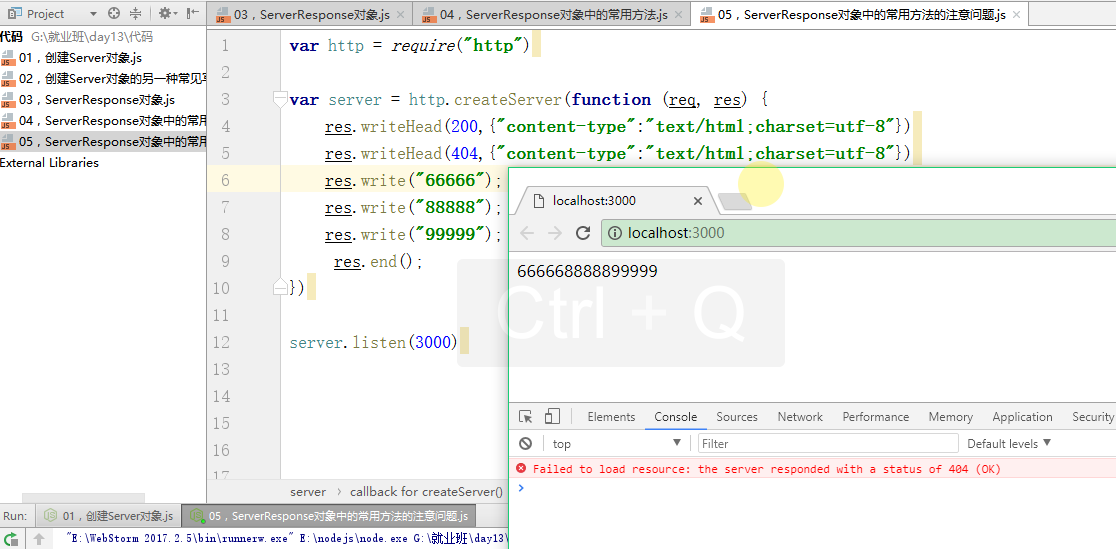
**1，write方法可以写多个：**



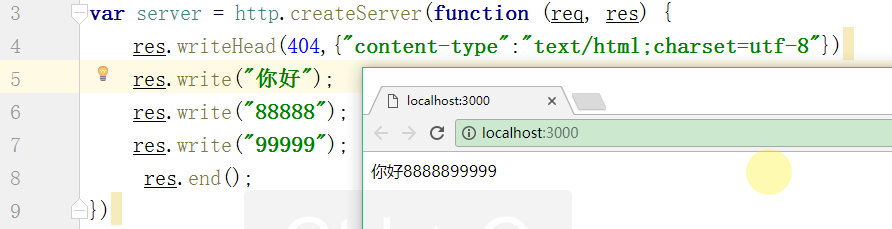
**2，end方法不能不写，不写的话，表示客户端一直在等待服务给响应**



**3，writeHead不能重复**



**4，writeHeader不能省略**

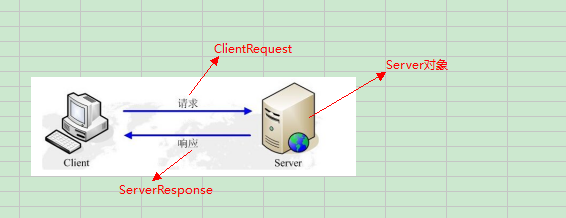


**5，在end这个方法中，可以写入响应正文**



## 7，IncommingMessage对象

看一个HTTP模型图：



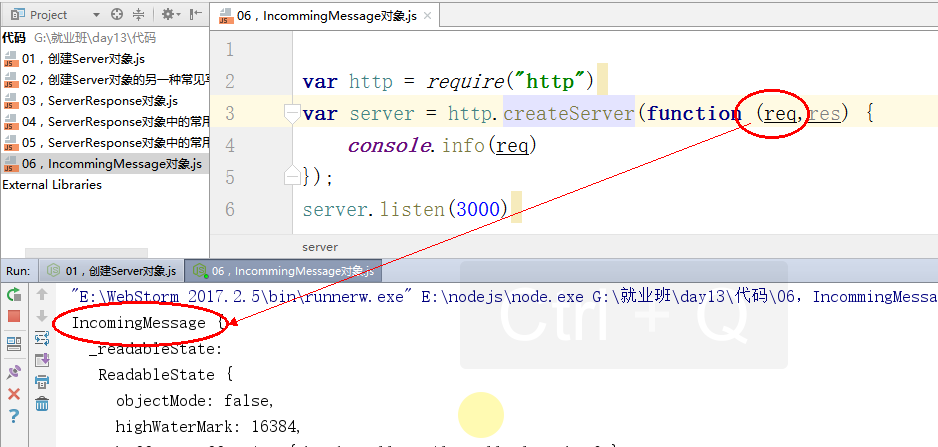
这里的IncommingMessage对象不是上面的模型图中的ClientRequeest对象。

对于监听器听第一个参数，它又是什么呢？



答：上面的req就是IncommingMessage对象。

代码如下：



至此，需要你知道，req就是IncommingMessage对象。

和SeverResponse对象一样，不需要手动创建，它是作为监听器的第一个参数存在的。

IncommingMessage对象都有哪些属性呢？

* httpVersion http的版本
* Headers 头信息
* rawHeaders 原始头信息
* **url 请求的url字符串 统一资源定位符是用来定位互联网的资源的。**
* Method 请求的方式 常见get和post
* Status 响应的状态码

Get: 在浏览器输入url img src a href form method get

Post: form method post

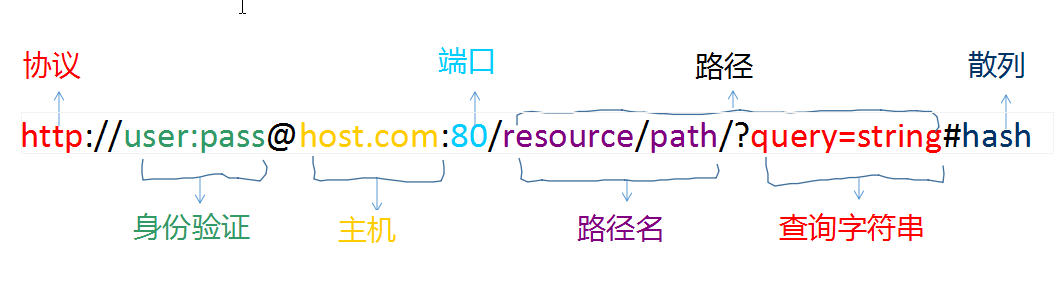
## 8，URL路由

### （1）URL介绍

URL Uniform Resource Location 统一资源定位符

URL用来表示一个资源（网页，音乐，图片，视频.....）在互联网上的一个地址。

在html中，href属性，以及img中的src都可以对应一个url。



<https://www.baidu.com/>

<http://news.baidu.com/>

<http://news.cctv.com/2017/11/28/ARTIFC9as9MixOGbte87V8mh171128.shtml>

https://guang.taobao.com/detail/index.htm?spm=a21bo.2017.2001.3.2aaa14f80QUeN&uid=2750874714&sid=8069062899&scm=1007.15939.89001.100200300000000&pvid=0ceac0c7-31f4-4d9f-8eb9-1699981008bf&itemid=551427655427

http https

user:pass 对于http协议，身份验证是不需要的

host 主机 服务器IP

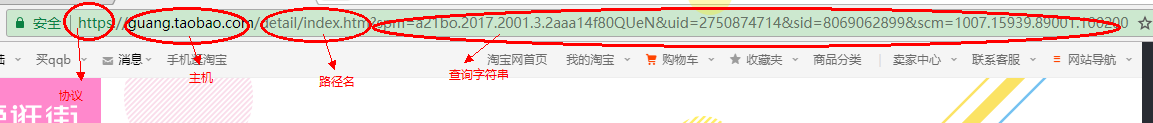
port 端口

path 路径名

query 查询字符串

& 用来连接多个查询字符串的

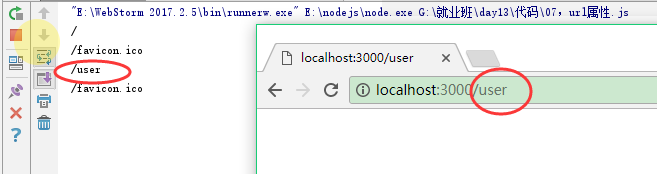
如下：



### 实现路由

利用req对象url属性，看代码：





当我们输入localhost:3000 结果是： /

当我们输入localhost:3000/user 结果是： /user

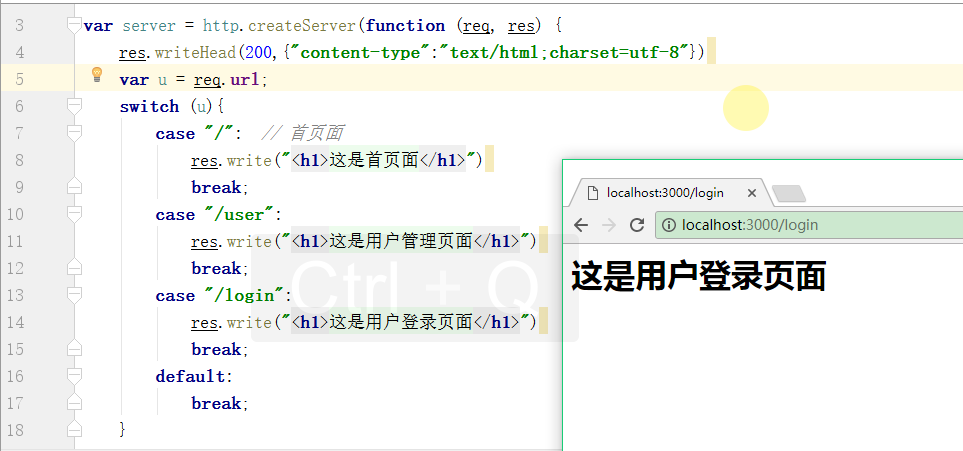
当我们输入localhost:3000/login 结果是： /login

当我们输入localhost:3000/list 结果是： /list

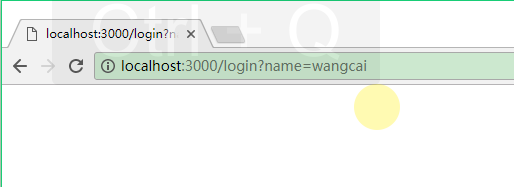
...

需求：当输入localhost:3000/user ，在页面上输出“这是一个用户管理页面”，当输入localhost:3000/login时，在页面上输出“这是一个登录页面”。

实现如下：



如果后面的查询字符串，结果如下：

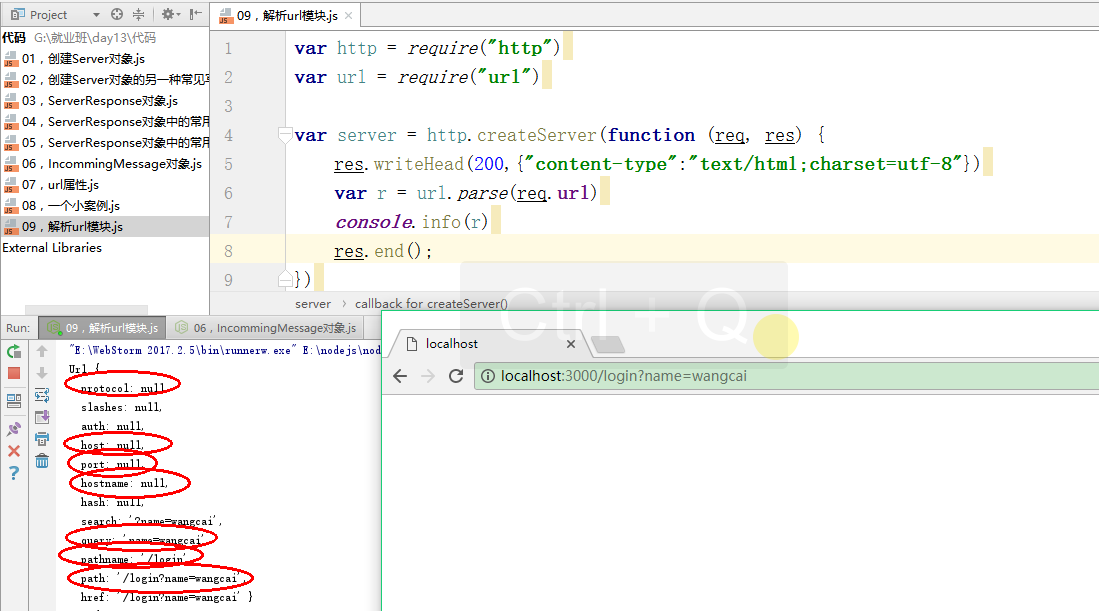


接下来就要对查询字符串进行解析，在node.js中专门提供了一个模块，这个模块叫url模块，专门用来解析url的。

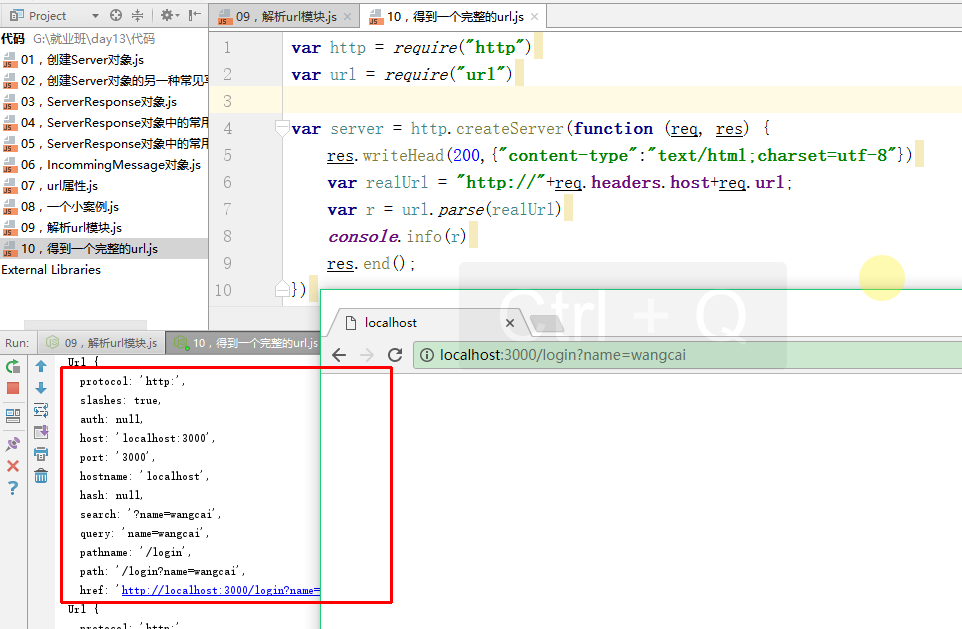
### url模块

Url模块也是一个系统模块，使用时，需要引入。

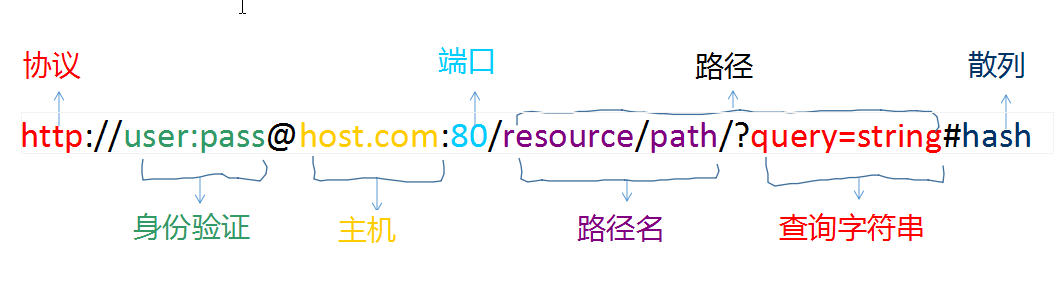
代码如下：



从上面的结果，可以看出，协议以及端口都是空，需求：得到一个完整的URL，怎么办？



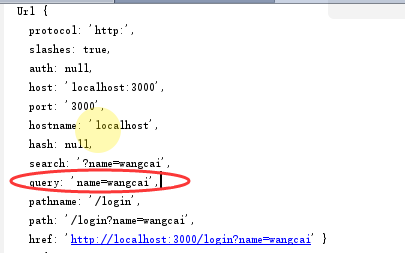
## 9，解析查询字符串



什么时候需要对查询字符串进行解析来？



解析查询字符串，就是去解析query:



在node.js中，专门有一个用来解析查询字符串的模块，叫querystring。

使用如下：

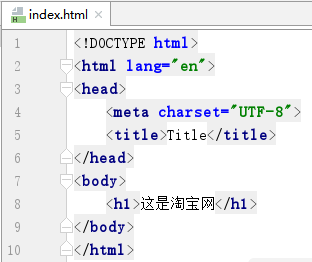


## 10，载入静态页面

前端人员把静态页面写完了，交给后端人员，后端人员就要考虑去把静态页面放到服务器上，就是载入静态页面。

在node中，是没有web容器的，所以在node中，需要写代码创建服务器，然后载入静态页面。

先写上一个静态页面，如下:



再写一个JS文件，载入静态页面，如下：



目前，服务器，已经OK了，使用浏览器访问之，如下：



接着，在index.html文件中，添加js, css, img.... 如下：



再次访问之，如下：



图片之所以没有起作用，是因为当读取完index.html后，在index.html中有css,js的代码，但是当看到img 标签时，它里面的src又会进行get请求，发现图片没有载入到服务器上，只是存在于本地，所以说，图片就不会起作用了。

最后，把js和css都使用外部引入，如下：



再去访问一下，如下：



**为什么呢？**

我们只读取了index.html，并没有读取js和css，img的内容。

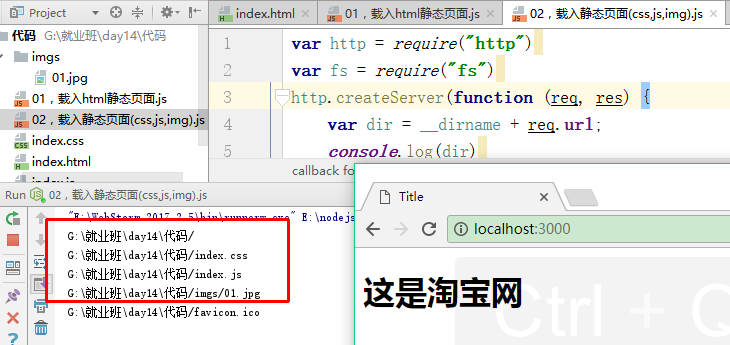
**解决呢？**

**第一步：如何获取请求信息呢？**

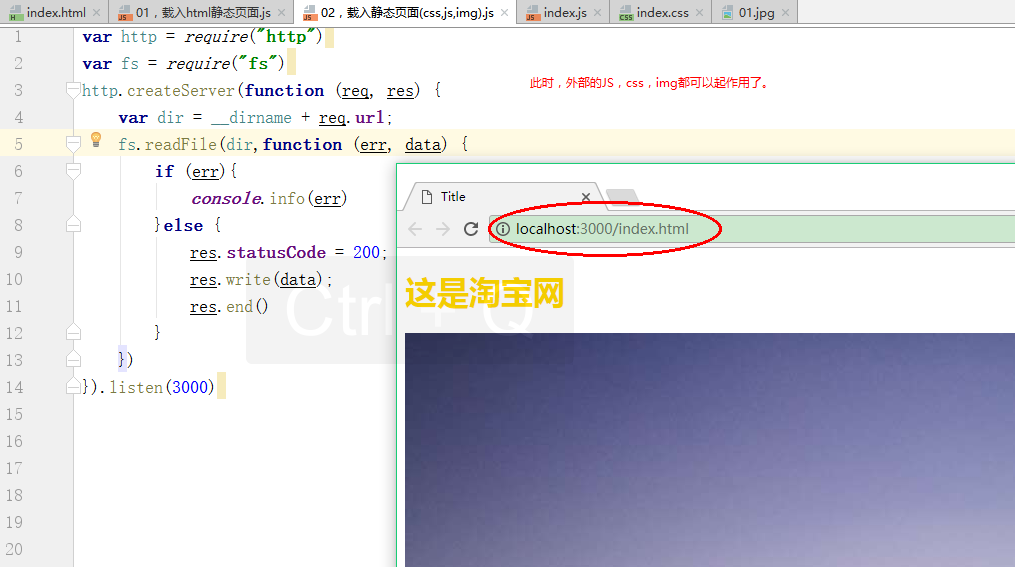
答：req.url



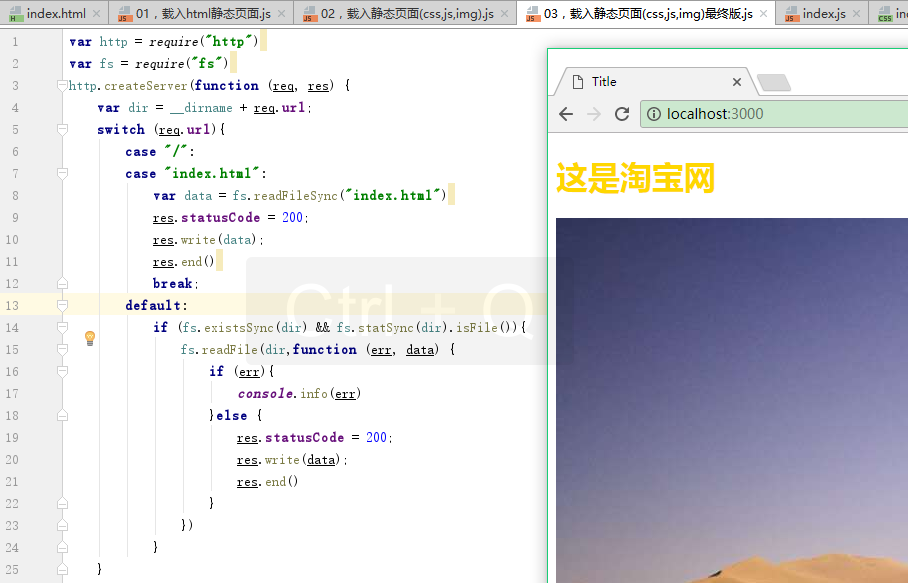
**第二步，得到上面的url的绝对路径**



**第三步，读取上面的所有的url，如下：**



还需要考虑，www.baidu.com时，我们没有在它后面加上index.html。如何不加Index.html就可以访问呢？ 代码如下：



上面的内容，也就是静态静态页面，非常重要。

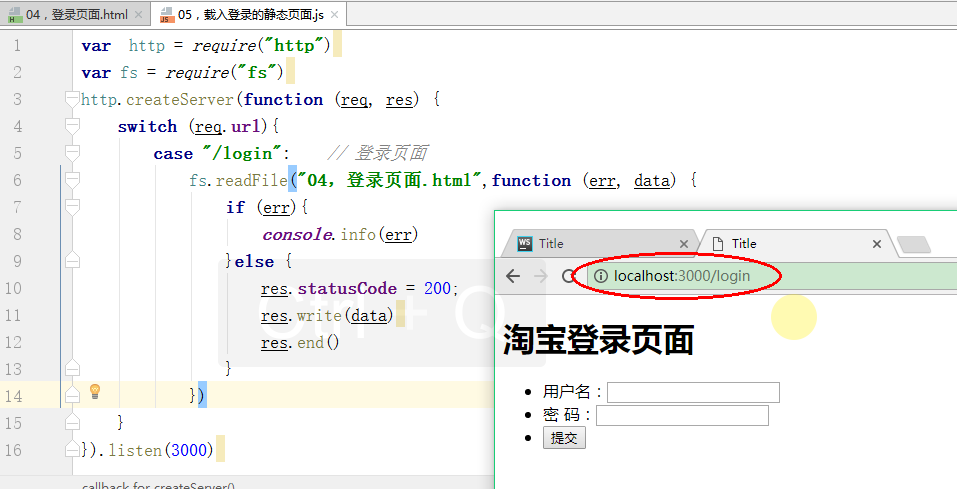
**把上面的内容，再次讲解！！！**

## 11，post请求以及响应

**第一步，实现post请求的静态页面，如下：**



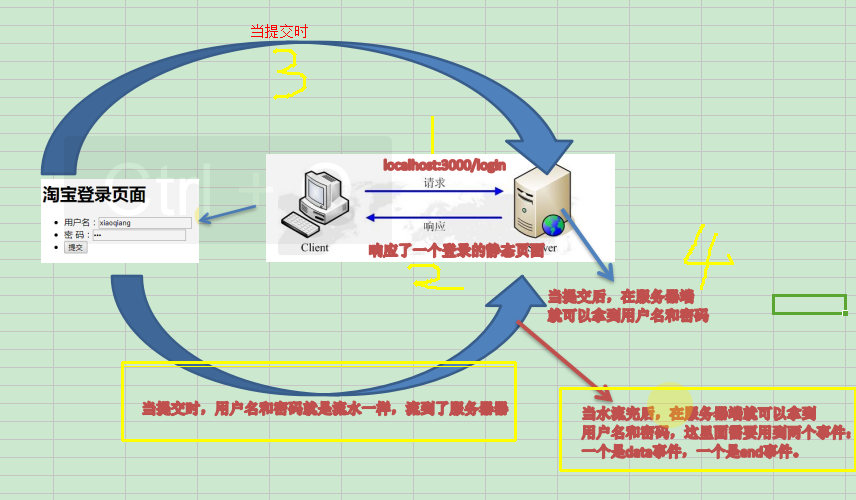
**第二步，载入上面的静态页面，如下：**



**第三步，我们将数据提交给某个地方，约定提交给/sigin 这个路由来处理。**



在/sigin中，就可以得到用户名和密码，对应的模型图，如下：

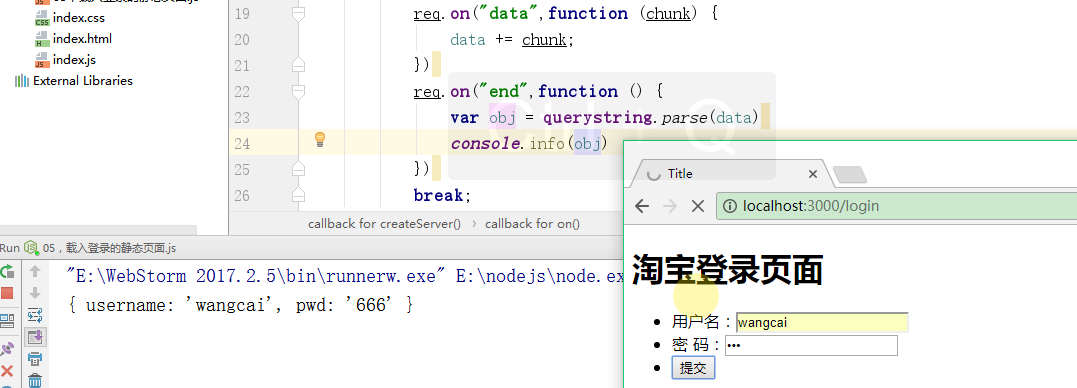


代码如下：

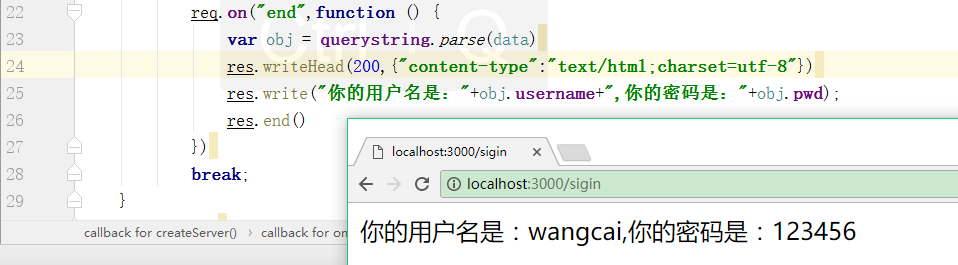


上面，我们拿到的是一个字符串，我们需要把上面的字符串，转化成对象。转化如下：

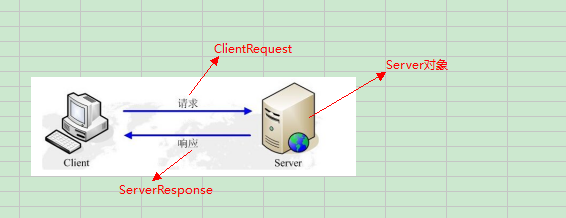
使用querystring模型：



通过上面的解析，就可以使用用户名和密码：



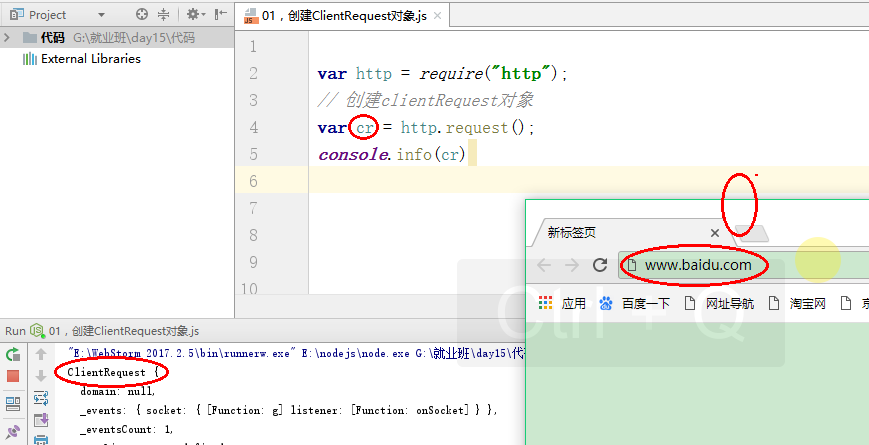
## 12，ClientRequest对象



在上面的模型图中，请求是浏览器发出的。当在地址栏中输入http://www.baidud.com时，当回车时，就会创建出一个ClientRequest对象。

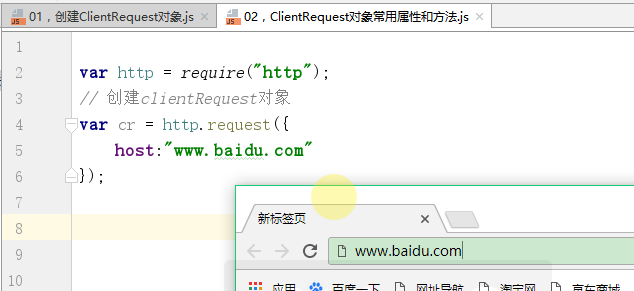
在nodejs中是可以创建出clientRequest这个对象的。好处：最常用的好处：写爬虫。

**第一步，创建出ClientRequest对象。**



**第二步，看一个，上面对象中有哪些属性和方法呢？**

* Write：把一个正文写入请求
* Host：请求服务器（IP地址）， 默认是localhost 本机的域名



**第三步，上面的创建出了这个请求对象，并且，设置请求的服务器，如何发送出请求呢？**

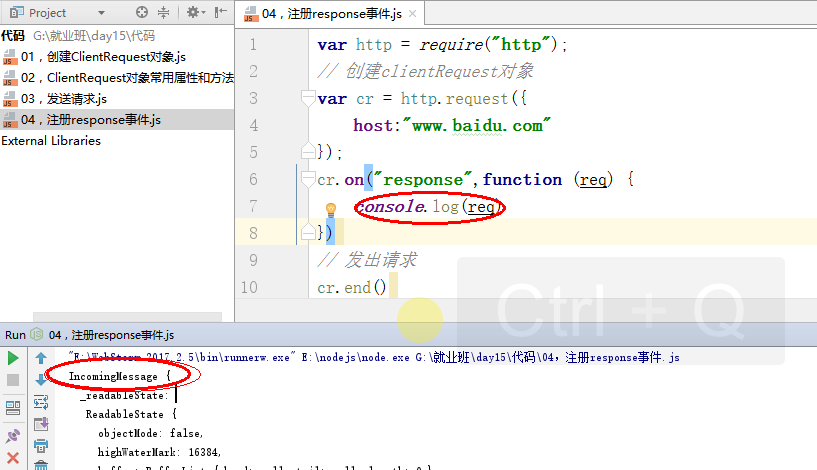
通过end方法



上面的代码，仅仅是把请求发送出去了，并没有写接收响应的代码。

**第四步，上面的请求已经发送了，怎么得到服务器的响应呢？**

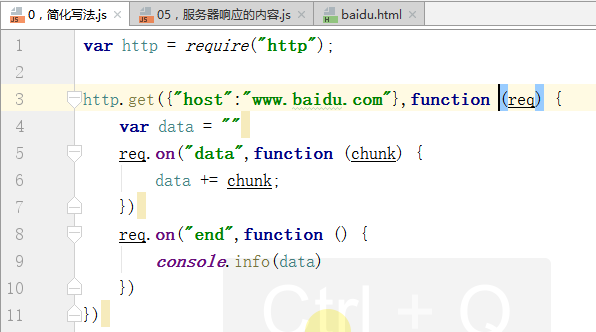
答：通过response事件，注册reponse事件。



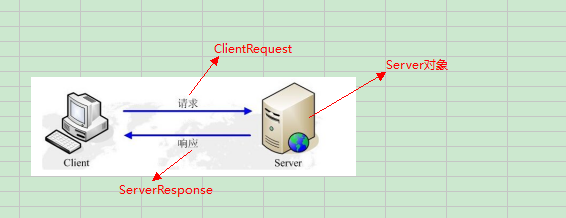
**第五步，在req中有两个事件，data，end事件**



**第六步，上面的写法，是为了让大家好理解，一般我们的写法如下：**



再次看一下，http模型图：



# 做一个留言板demo

## npm

npm : node package manger node的包管理器

**模块：**一个文件就可以看作是一个模块

**包：**在项目开发中，一个模块有时候不足以完成某个功能，这个时候要多个模块同共完成，当把多个模块放在一起时，就形成了一个文件夹，也就是包

当安装node时，npm这个工具就已经安装好了。

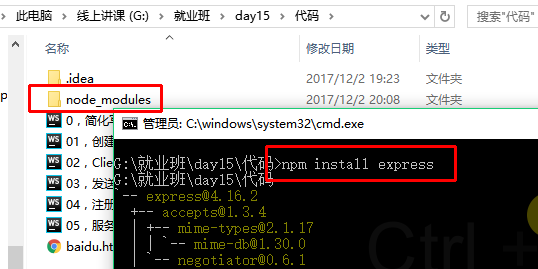
说白了，npm使用的话，也就是记一些命令。

**常用命令：**

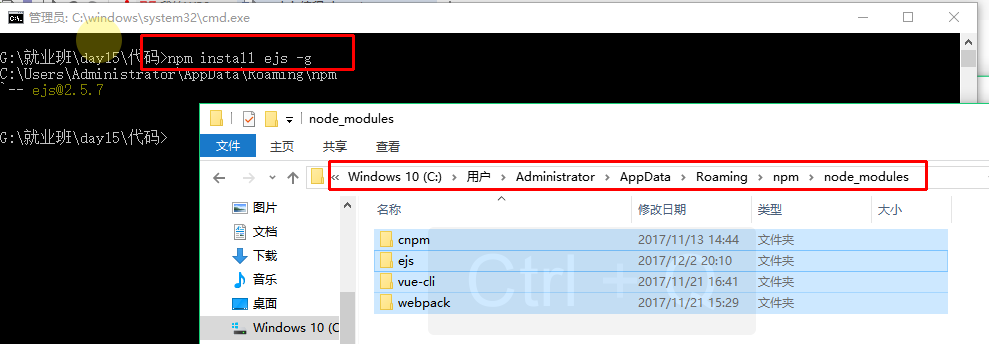
**npm help**



**npm install 包名 安装包的 本地安装**



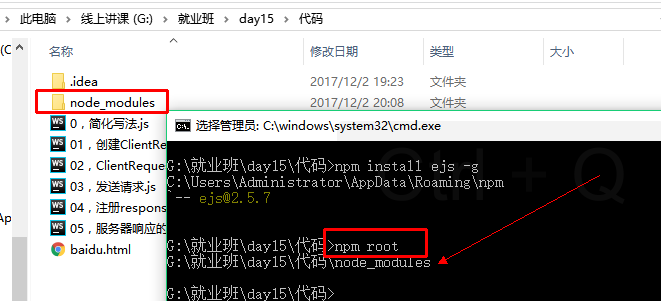
**npm install 包名 -g 全局安装**



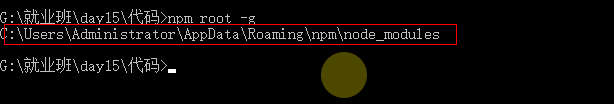
**npm uninstall 包名 卸载包**

**npm update 包名 更新包**

**npm root 查看当肖包的安装路径**



**npm root -g 查看全局包的安装路径**



## ejs模板引擎

在前面写的页面中，里面的数据都是假的，或死数据，通常页面中的数据是从数据库中取出来的，拿到数据后，需要把数据放到页面中，就可以通过ejs模板引擎来实现。

**作用：将数据成html模板进行分离。**

在一个页面中，数据部分通常是JSON，在HTML文件（模板文件）, 需要通过特定的语法，把真数据放到模板中。

**在EJS中，特定的语法，如下：**

* 执行语句：<% %>
* 显示语句：<%= mes %>
* include语句：<% include 模板文件名 %>

**如何将模板与数据结合在一起呢？**

* render 针对数据
* renderFile 针对文件

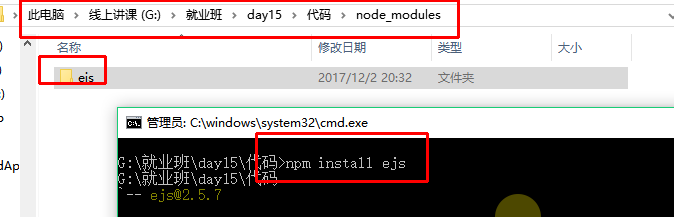
**写一个模板文件，也就是HTML文件，如下：**



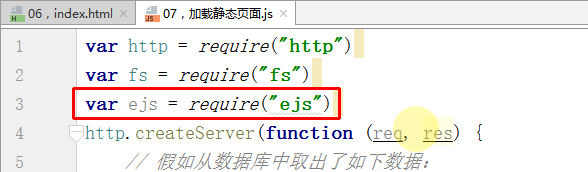
**后端人员拿到上面的前端人员写的模板文件后，首先，它会加载静态页面，如下：**



接下来，需要把数据部分与模板部分进行结合，要使用ejs模板引擎，第一步，肯定是下载这个包。



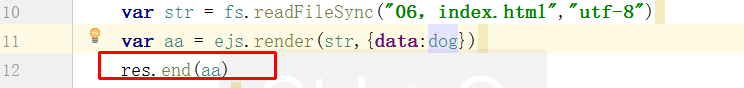
在项目， 要引入ejs这个包，引入完之后，才能使用：



最后，把数据和模板进行相结合，如下：



注意，需要把响应的内容，给浏览器响应回去：

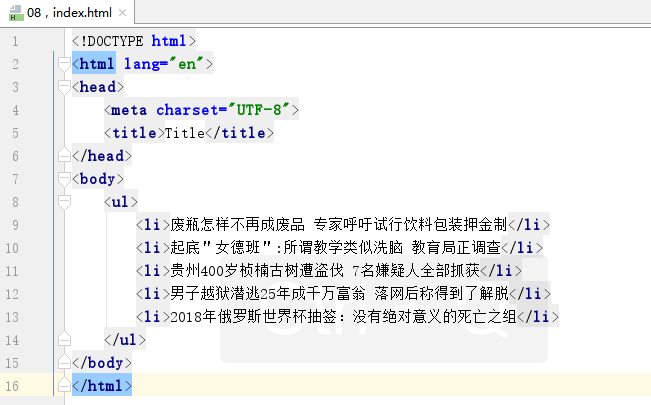


结合后，就可以把原先模板中的死数据，换成真实的数据，如下：



## EJS案例

HTML模板如下：

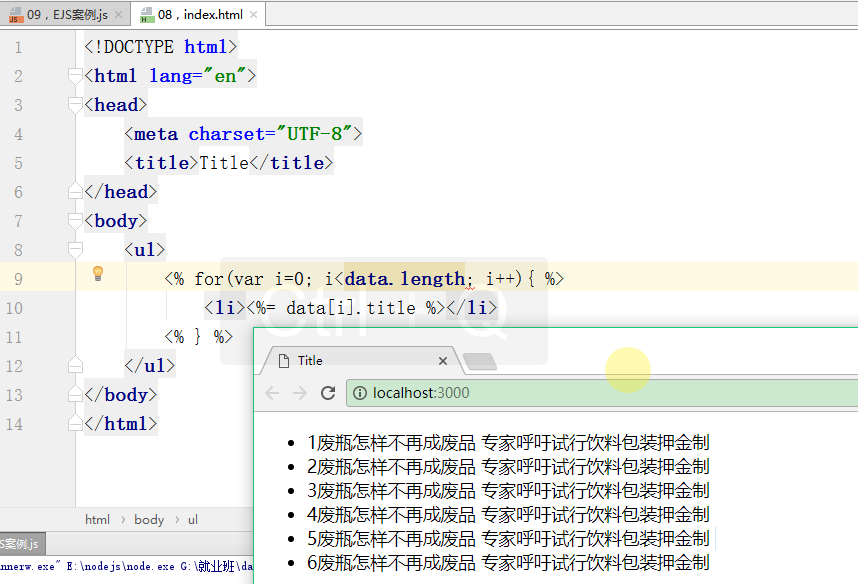


前端人员写好模板后，将模板，交给后端进行处理：





**接着，需要把上面的假数据替换成真实的数据**



## 需求分析

**有两个功能：**

1. 发表留言
2. 显示留言

**数据保存在什么地方？**

按理说数据都是保存在数据库中的，由于我们没有学习数据库，所以说，目前，我们只把数据保存在文件中。

**模板技术？**

**Ejs**，Jade， .....

**项目使用的技术栈：**

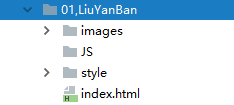
后端：node+ejs

前端：HTML+CSS+JS

## 功能实现

### （1）第一步，前端人员会把写好的静态页面，给后端人员，在node中，后端人员首先要加载静态页面。

如下是前端人员给后端人员的静态页面。



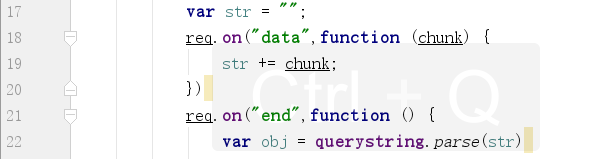
创建服务器，加载上面的静态页面，如下：



### （2）第二步，当点击提交按钮时，我们需要把数据提交给服务器，并且将数据保存到一个文件中

**用到的知识点：**

需要用到两个事件：data , end



上面的代码中，我们就得到了从客户端提交给服务器的数据，并且把数据转化成了一个对象 。

接着，需要把一个对象保存到一个文件中， 对于一条留言， 就是一个对象，由于提交过来的数据没有时间，所以自己定义一个对象，把提交过来的数据，封装到自己创建的对象中，如下：



上面的代码，我们就拿到了一个留言对象，在项目中，肯定不是一条留言，肯定是多条，肯定是多个对象，多个对象的话，我们需要把对象放到数组中，如下：



此时，数组中就有了一条留言。第二次，又需要把一个对象放到一个数组中，此时，数组中就有两个留言了，有了留言，接下来，就需要把数组转化成字符串，覆盖式写文件，留言写到一个文件中，如下：

注意：把JSON转化成字符串，使用JSON.stringfy(arr)

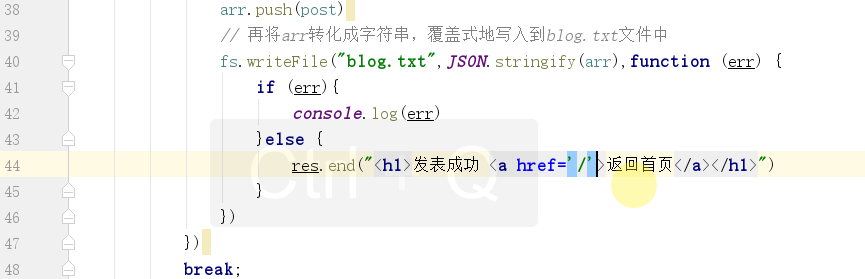


如果blog.txt存在的话，先取出blog.txt中的内容，转成对象，放入到数组中。如下：

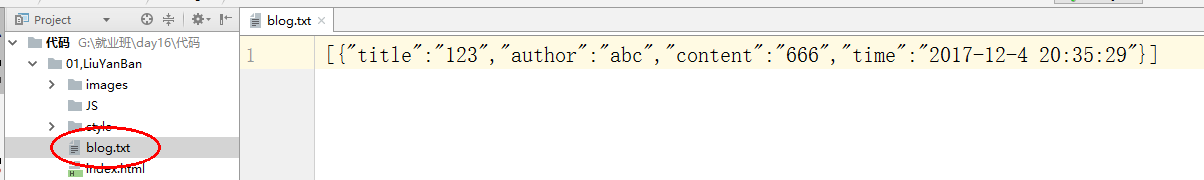


完整代码如下：

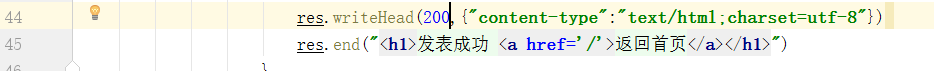




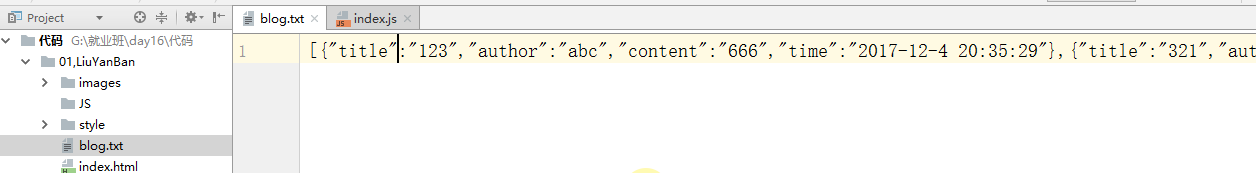
运行之，如下：



小问题， 响应的时候，需要设置响应的头信息，如下：



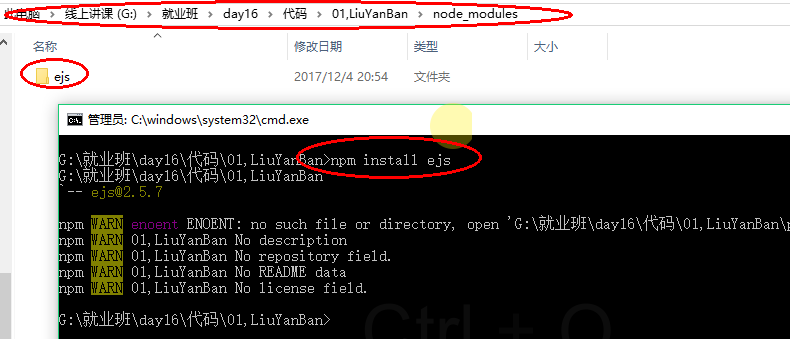
再次运行之，如下：

、

### 首页面显示留言

思路：当读取首页面时，不能直接把首页面给响应回去了，需要把数据和首页面进行绑定，进行渲染。

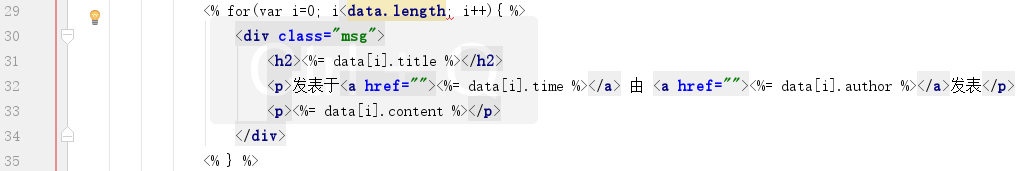
此时，用到了ejs，所以需要安装ejs，如下：



代码如下：



需要在index.html中通过ejs的特定语法，完成真实数据的输出，如下：



# 六，ES5和ES6中的新语法

**问一个问题：ES6中的新语法为什么不在讲JS基础时讲呢？为什么放在讲node时，讲呢？**

ES5或ES6中新增的方法，在客户端JS中，不同的浏览器有不同的兼容性。但是在服务器端，不存在兼容性，所以把上面的新方法放到服务器编程中说。

在讲JS基础时，数组和字符串中有很多的方法。

## ES5开始新增的一些数组相关的方法

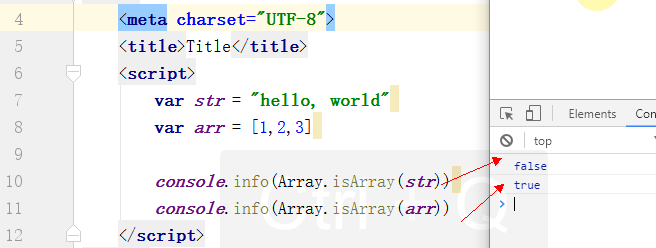
* array.isArray
* forEach()
* map()
* filter()
* some()
* every()

### Array.isArray

Array叫构造器， 不是一个对象。 前面点语法，都是对象.方法/属性。这里的Array是构造器，也可以使用点语法，后面跟的这个方法叫静态方法。一般是使用构造器进行调用的，不是对象进行调用。

上面的如果不好理解，没有关系，等到讲到JS高级时，都可以理解了。

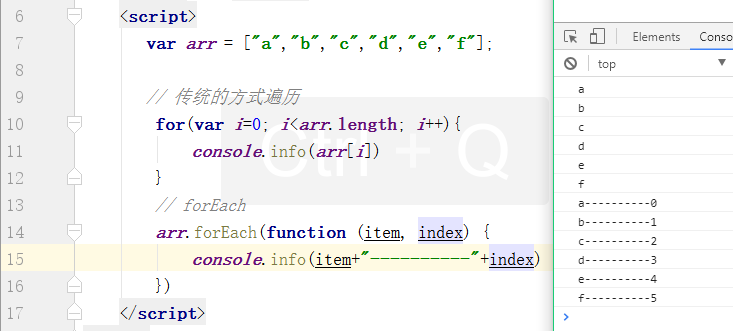
代码如下：



### forEach()

作用：用来遍历数组

代码如下：



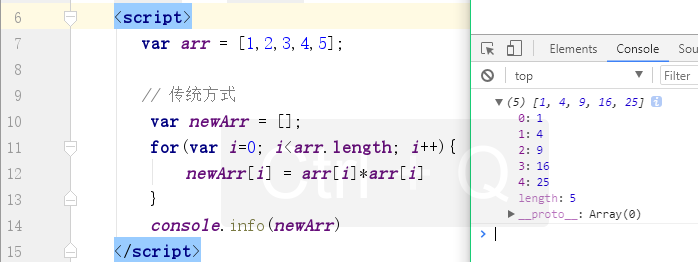
### map()

原意：地图，映射

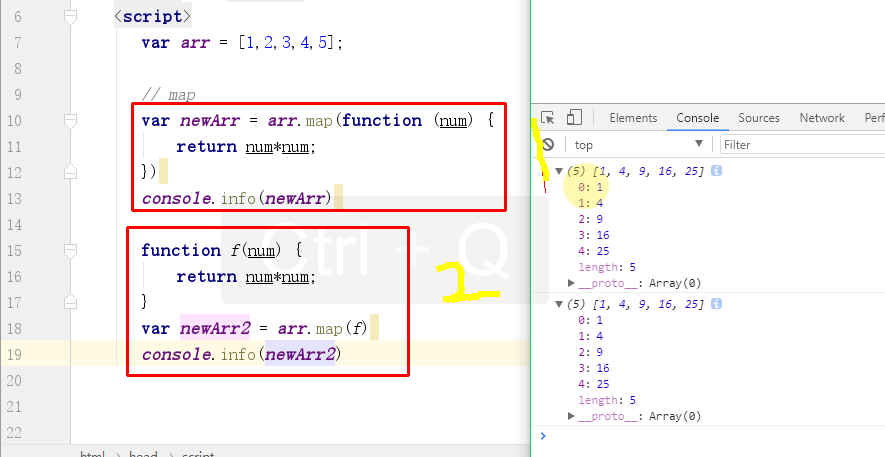
map方法个由原数组中的每个元素调用一个指定的方法后的返回值组成的一个新数组。

案例：对于个数组arr=[1,2,3], 求里面每一个元素的平方，放到一个新数据中。

**传统的方式实现：**



**Map的方式：**

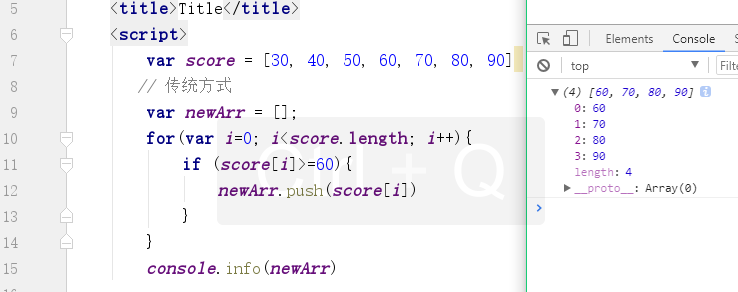


### filter()

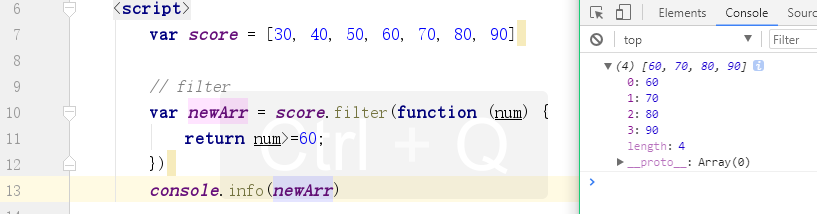
filter对于一个数组，运行某个规则，过滤掉某些元素，返回剩余的元素，剩余的元素组成一个新数组。

案例：对于一个数组 var score = [30, 40, 50, 60, 70, 80, 90] 把及格的分数找出来

**传统的方式，如下：**



**filter:**



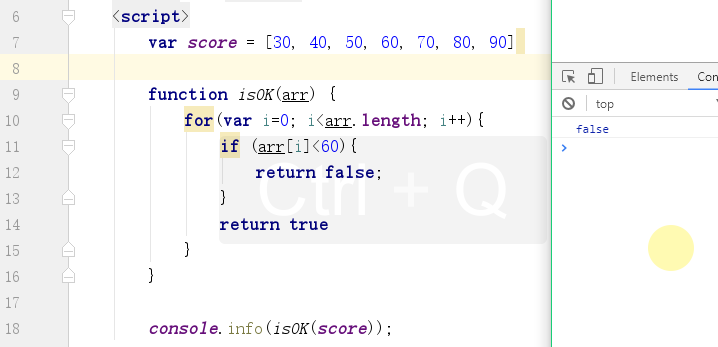
### Every()

every是所有的意思

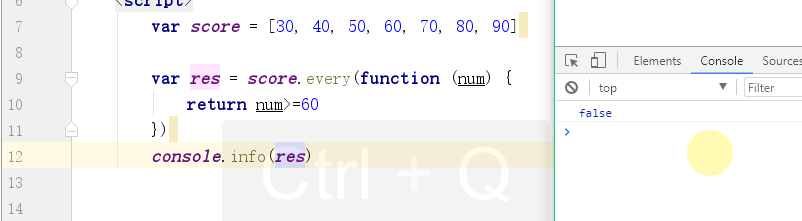
作用：判断数组中每一个元素是否满足某个规则，类似于&&

**需求： var score = [30, 40, 50, 60, 70, 80, 90] 数组中的分数是否都及格了。**

传统方式：

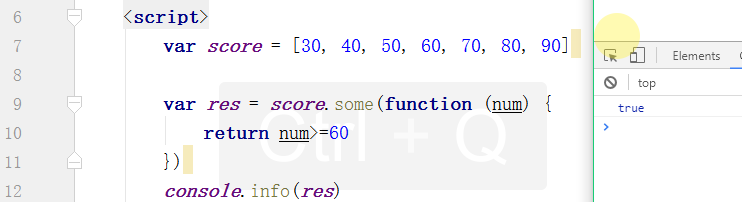


Every:



### （6）some()

有点类似于 ||



## 2，新语法之let和const

前面，声明一个变量，使用的是var这个关键字，有哪些特点：

* 声明提升
* 不能被删除
* 变量的值可以修改
* 可以先声明，不赋值，后面再赋值
* 使用var可以重复声明

在JS中，是没有块级作用域，只有函数作用域和全局使用域

如果让一门编程语言非常强大的话，只有变量没有常量，是不能满足的。所以在ES6中，就提供了const let

### （1）const 用来定义一个常量的

**代码如下：**

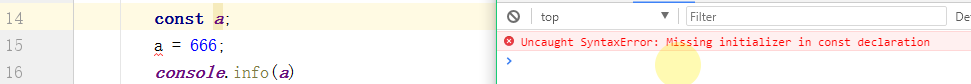


Const定义的常量有如下的特点：

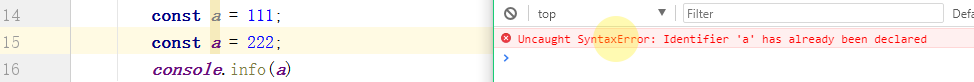
**1，一经定义，不能修改：**



**2，常量必须在声明时初始化**



**3，常量不能重复**



1. **const常量没有提升的功能**



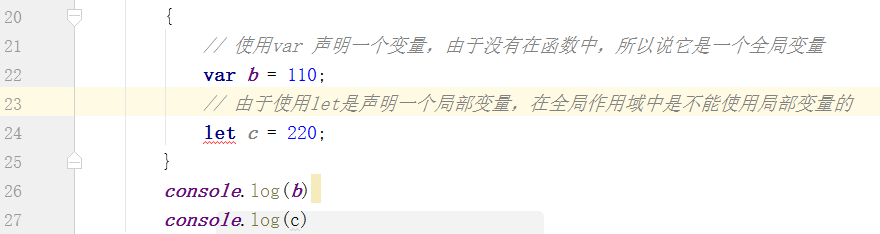
**在写代码时，什么时候用const， 什么时候用var?**

1. 如果你需要一个会变的量，就用var

2，如果你需要一个不变的量，就用const

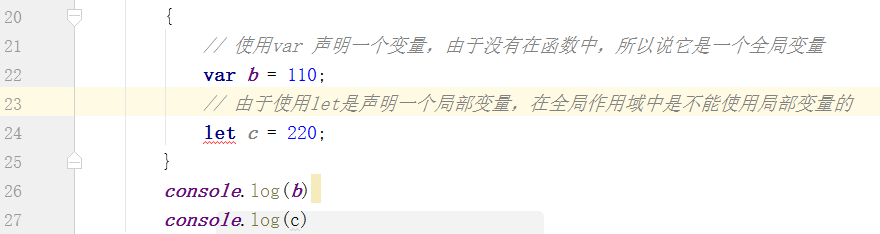
### Let 的用法

let表示声明一个局部变量。在JS中，全局变量和局部变量的分界点是函数。



let声明的变量有如下的特点：

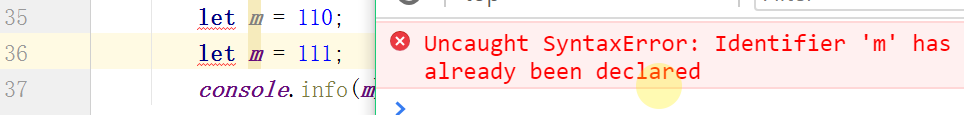
1. **只在局部作用域中起作用。局部作用域：函数， {}**



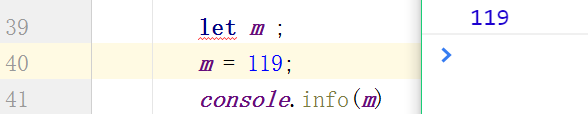
1. **没有提升的功能**



**3，不能重复声明let变量**

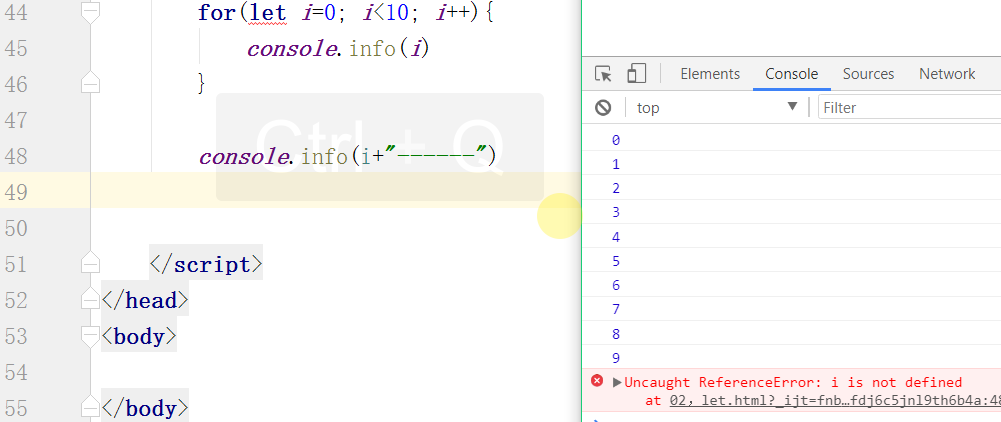


**4，let声明的变量可以先声明，不赋值，后面再赋值**



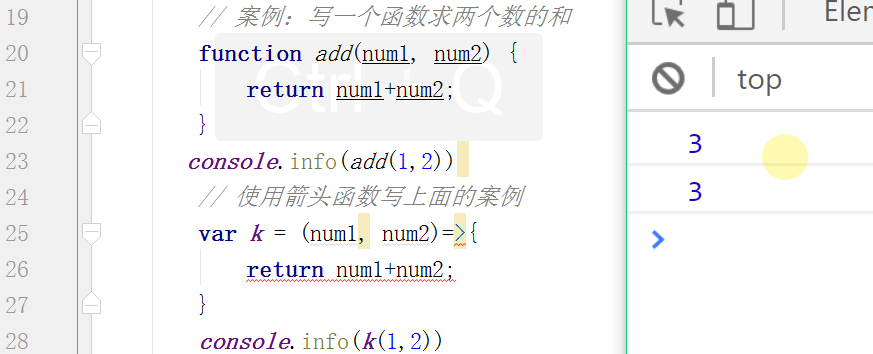
**在什么时候使用let声明一个变量呢？**

目前，最常见是在for循环中，如下：



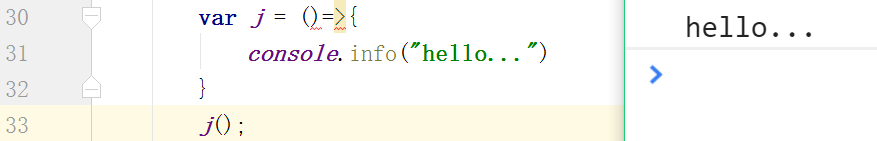
## 3，箭头函数

在以前，函数有两种方式：函数声明，函数表达式，如下：

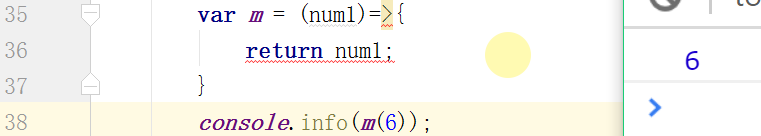


根据参数的不同，还有如下几种写法：

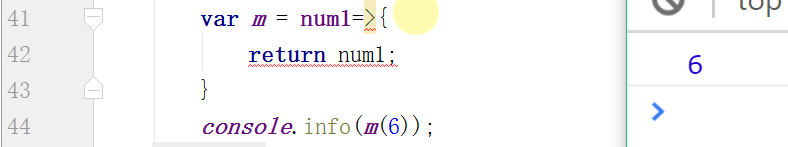
**第一种，0个参数时，如下：**



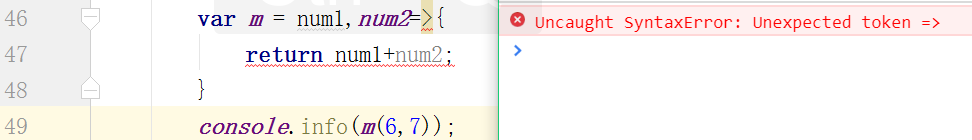
**第二种，1个参数时，如下：**



**当有一个参数时，可以把()省略不写，如下：**



**第三种，如果说有多个参数时，就不能省略小括号了，如下：**



## 4，JS都有哪些特性？以及JS中的一切皆对象

JS：

JS的基础

DOM：客户端JS

Node: JS在服务器端的运行环境

JS是一门运行在浏览器端的脚本语言。

JS不只可以运行在浏览器端，也可以运行在服务器端。

**特性：**

1. 弱数据类型。体现：定义个任何数据类型的变量都使用var，还有对形参设置，不需要指定类型。
2. JS是解释性的语言，不是编译性的。

**JS中的数据类型：**

基本数据类型：number, string, boolean, undefiend, null

引用数据类型：object, array, function

**JS中一切皆对象**

**证明：**基本数据类型是对象，引用数据类型也是对象 ----> **一切皆对象**

**JS中的对象分类：**

1. 内置对象 是ECMAScirpt中的对象
2. 宿主对象 DOM与BOM中的对象
3. 自定义对象

**JS不是一门面向对象的编程语言，它仅仅是一门基于对象的编程语言，传统的面向对象的编程语言有三大特征：封装，继承，多态。在JS中对于三大特征都是模拟出来的。**

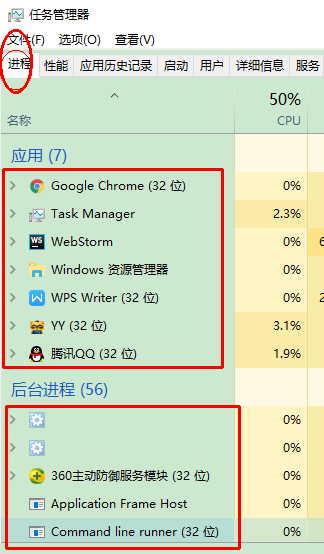
## 5，什么是单线程

**什么是进程？**

当一个软件运行时，就是一个进程。

进程（Process）是计算机中的程序关于某数据集合上的一次运行活动，是系统进行资源分配和调度的基本单位，是[操作系统](https://baike.baidu.com/item/%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E8%BF%9B%E7%A8%8B/_blank)结构的基础。

如下：



**在一个软件运行过程中，真正干活的是线程，一个进程中可能包含若干个线程。**

**什么是单线程？**

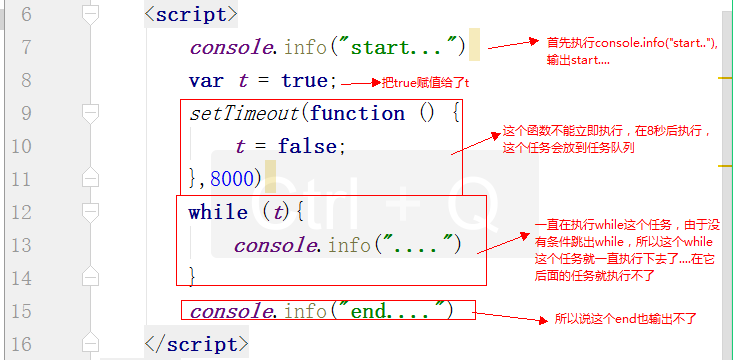
如果有多个任务时，任务只能一个一个的执行。

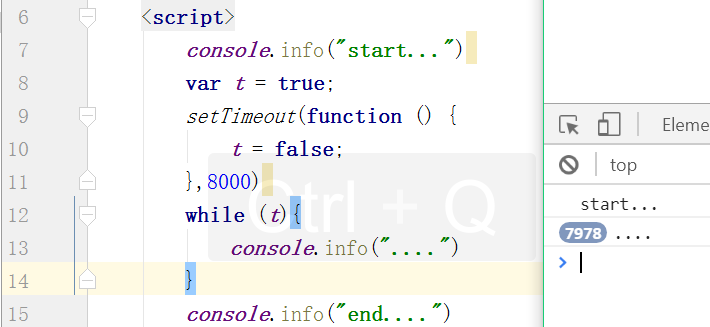
**餐厅：**

有多个服务员。---> 多线程 每个服务员都可以干活。

有1个服务员。---> 单线程 这个服务员要服务好多的餐桌。

代码如下：





**单线程优缺点：**

优点：实现比较简单，任务一个一个

缺点：只要有一个任务耗时比较长，后面的任务必须进行排除等待，会拖延整个程序的执行