复习

模块分类

|  |  |  |
| --- | --- | --- |
|  | 以路径开头 | 不以路径开头 |
| 文件模块 | require('./circle.js')  自定义模块 | require('url')  核心模块 |
| 目录模块 | require('./mydir')  自动寻找index.js文件  package.json文件寻找main属性对应的文件 | require('mysql');  到node\_modules中寻找目录名称为mysql的目录 |

npm install 包名称;

querystring parse() stringify()

url parse() format()

protocol hostname port pathname query

fs fs.stat/fs.statSync fs.mkdir/mkdirSync

rmdir/rmdirSync readdir/readdirSync

readdir(path, callback) var res=readdir(path)

writeFile(path,data,callback)

1.文件系统模块——fs

①fs.unlink(path,callback)/fs.unlinkSync(path) 删除文件

②fs.existsSync(path) 判断文件是否存储

存在true 不存在false

练习：判断文件num.txt是否存在，如果不存在则创建，并初始化数字0；在原来的数字上加1。

③fs.readFile(path,callback)/fs.readFileSync(path) 读取文件

返回的数据是buffer形式。

④fs.appendFile(path,data,callback)

fs.appendFileSync(path,data)

追加写入，如果文件不存在则创建文件，如果文件已经存在，则在末尾写入数据。

练习：使用文件操作来创建文件user.txt，每次写入一个对象,

{uid:1, uname:'tom', upwd: '123456'}

2.http协议

是浏览器和web服务器之间的通信协议。

(1)通用头信息

Request URL: 请求的URL，要向服务端请求哪个文件。

Request Method: 请求的方法 get/post

Status Code: 响应的状态码

2\*\* 服务器成功的响应

3\*\* 响应的重定向，跳转到另一个网址

4\*\* 客户端错误

5\*\* 服务器端错误

Romote Address: 请求的远程服务器的IP地址和端口

(2)响应头信息

Connection: keep-alive；连接的方式：持续连接

Content-Type: text/html; 响应的文件类型

Content-Encoding: 响应的文件压缩形式

Transfer-Encoding:响应时的传输方式，chunked(分段传输)

Location: 响应时跳转的URL，通常结合着300系列状态码。

(3)请求头信息

Accept: 客户端接受的文件类型有哪些

Accept-Encoding: 客户端接受的文件压缩形式

Accept-Language: 客户端接受的语言类型

Connection: 客户端和服务器的连接方式，持续连接

(4)请求主体

可有可无，客户端向服务器端传递数据

3.http模块

可以模拟浏览器向服务器端发请求，也可以创建web服务器

(1)模拟浏览器

http.get(url, callback)

get 请求的方法

url 请求的网址

callback 回调函数，用来获取服务器端的响应

res 响应的对象

res.statusCode 获取响应的状态码

res.on('data', (buf)=>{ })

使用事件来获取服务器端响应的数据

buf是服务器端响应的数据，格式为buffer数据。

(2)创建web服务器

var server=http.createServer() 创建web服务器

server.listen(3000) 分配端口，监听3000端口的变化

server.on('request', (req,res)=>{ });

//接收浏览器的请求，是一个事件，一旦有请求，自动执行

req 请求的对象

url 请求的路径，显示端口后的部分

method 请求的方法，直接通过地址栏默认使用get方法

headers 请求的头信息

练习：创建web服务器，监听3001端口，使用事件监听浏览器的请求，打印请求的方法、URL；

http://127.0.0.1:3001/admin/login.html

http://localhost:3001/member/shopping.html

05\_server.js

res 响应的对象

write() 响应的内容为文本形式，向浏览器中写入文本。

writeHead( 302,{ } ) 设置响应的状态码和响应的头信息；如果要跳转需要设置 Location属性。

end() 响应结束

练习：创建web服务器，监听3000端口；接收浏览器端的请求； 06\_server.js

/login 响应文本 this is login page

/member 响应文本 this is member page

/ 跳转到 /member

如果以上都没有匹配的， 响应文本 404 not found

3.express框架

基于NODEJS，用于构建web服务器的框架

官网：www.expressjs.com.cn

安装：npm install express

|  |
| --- |
| const express=require('express');  var server=express();  server.listen(3000); |

(1)路由

浏览器向web服务器发来请求，web服务器要根据请求的方法和请求的URL来作出响应。

路由三要素：请求的方法、请求的URL、响应的内容

响应的对象(res)

res.send() 响应文本，只能响应一次send；如果是数字认为是状态码。

res.sendFile() 响应文件，必须使用绝对路径(\_\_dirname)

res.redirect() 响应的重定向

课后任务

(1)复习今天内容，整理思维导图

(2)练习

使用express创建web服务器，创建以下路由

get '/index' 发送文本"这是首页"

get '/login' 发送文件 "login.html"

post '/register' 发送文本"注册成功"

get '/' 跳转到 /index

(3)预习express的中间件，浏览器向服务器传递数据方式有哪些(post，查询字符串，路由传递)