复习

模块

|  |  |  |
| --- | --- | --- |
|  | 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('./02\_2')  会找目录下package.json中main属性对应文件，找不到引入index.js | require('mysql')  到当前目录下node\_modules中寻找mysql目录；  引入第三方模块 |

npm install mysql

querystring 查询字符串

https://www.jd.com/search?kw=电脑

将查询字符串转对象 parse

把对象格式化为查询字符串 stringify

url

parse/format

全局函数

setTimeout/clearTimeout

setInterval/clearInterval

setImmediate/clearImmediate

process.nextTick

fs模块

同步(sync)和异步

同步：等待前边的代码执行完，才能执行后边的代码；会阻止后边代码的执行，通过返回值来获取结果。

异步：不会阻止后边代码执行，后边的操作可以先执行，不必等待前边的操作结束，通过回调函数来获取结果。

1.fs模块

(1)查看文件状态

fs.stat(path, callback) / statSync(path)

(2)创建目录

fs.mkdir(path, callback) / mkdirSync(path)

(3)移除目录

fs.rmdir(path, callback) / rmdirSync(path)

使用异步移除目录mydir

使用同步移除目录mydir2

(4)读取目录

fs.readdir(path, callback) / readdirSync(path)

callback

err 可能产生的错误

result 读取的结果，格式为数组

(5)创建文件/写入文件

fs.writeFile( path, data, callback )

/ writeFileSync(path,data)

data要写入的数据

如果文件不存在，会创建文件并写入数据；

如果文件已经存在，会清空文件中内容，然后写入数据

(6)追加写入文件

fs.appendFile(path, data, callback) /

appendFileSync(path, data)

如果文件不存在，会创建文件，然后写入数据

如果文件已经存在，会再文件的末尾写入数据

练习：创建数组，每个元素是一个课程名称，遍历数组，分别获取元素，将数据使用**同步方法**写入到文件data.txt

(7)读取文件

fs.readFile(path, callback) / readFileSync(path)

callback

err

data 读取的数据，格式为buffer

(8)删除文件

fs.unlink(path, callback) / unlinkSync(path)

(9)判断文件是否存在

fs.existsSync(path)

练习：判断data.txt是否存在，如果存在，则删除

(10)文件流

var readStream=fs.createReadStream(文件路径) //创建可读流

readStream.on('data', function(chunk){ })

当有数据流入，自动触发事件；

通过回调函数来获取 ，chunk就是获取的数据流

readStream.on('end', function(){ })

当读取结束，自动触发事件

pipe 管道，可以将数据从可读流添加到可写流

2.http协议

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,对应浏览器地址栏；向服务器获取哪些内容

Request Method: 请求的方法，获取内容的方法 get/post

Status Code: 响应的状态码

1\*\*：正在响应，还没有结束

2\*\*：成功的响应

3\*\*：响应的重定向，跳转到另一个URL

4\*\*：客户端请求错误

5\*\*：服务器端错误

Remote Address：请求的服务器的IP地址及端口号

(2)响应头信息

Connection: 连接方式，keep-alive 持久连接

Content-Encoding: 内容压缩形式， gzip

Content-Type: 响应的文件类型

Location: 跳转的URL,常配合着状态码3\*\*使用

(3)请求头信息

Accept: 浏览器接收的文件类型有哪些

Accept-Encoding: 接收的压缩形式有哪些

User-Agent: 客户端使用的浏览器版本

(4)请求主体

可有可无，浏览器向服务器发请求传递的数据

3.http模块

模拟浏览器向web服务器发请求，还可以创建web服务器

(1)模拟浏览器

http.get(url, callback) 发送请求

url 请求的URL

callback 回调函数，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', function(chunk){ })

事件: 监听服务器端是否有数据传输过来

chunk 就是传输的数据，格式为buffer

(2)创建web服务器

|  |
| --- |
| var app=http.createServer(); //创建web服务器  app.listen(8080);//设置监听的端口  //监听浏览器的请求  app.on('request', function(req,res){  req 请求的对象  req.url 请求的url，获取端口号后边的部分  req.method 请求的方法  req.headers 请求的头信息  res 响应的对象  res.write() 设置响应的内容  res.writeHead( 状态码, 头信息对象 )  res.end() 结束并发送响应到浏览器  }) |

课后任务

(1)复习今天内容，整理思维导图

(2)练习

创建web服务器，监听端口8080，接收浏览器的请求，根据请求作出不同的响应 http://127.0.0.1:8080/reg

/login 响应内容 this is login page

/reg 响应内容 this is reg page

/study 跳转到 http://www.tmooc.cn

（3）预习express