1、同步和异步。

同步：阻止后续代码执行，先执行完当前的同步操作才能往后执行，通过返回值获取结果

异步：不会阻止后续代码执行，在一个独立的线程运行，执行完以后把结果放入回调函数，通过回调函数获取结果。

2、fs模块

statSync('文件路径') stat('文件路径',回调函数)==>回调函数用于获取文件状态。

fs.stat('./test.txt', (error, result) => { //error 可能产生的错误

if (error) {

throw error

} else {

console.log(result)

}

})

3、读取目录。

readdir('目录路径',回调函数) / readdirSync('目录路径')

结果是一个数组。得到目录下所有文件和目录。

fs.writeFileSync(path1,str);

//如果文件不存在，会先创建文件，如果文件已存在，会覆盖原来文件，写入新的数据

fs.writeFile(path,data,回调函数);

读取文件内容

fs.readFile(path,回调函数) / fs.readFileSync(path) 读取的文件数据格式为Buffer

向文件中写入内容

fs.writeFile(path,data,回调函数) / fs.writeFileSync(path,data)

如果文件不存在，会先创建该文件，然后向文件中写入内容，文件已存在，会覆盖当前文件然后写入内容。

向文件中追加内容

fs.appendFile(path,data,回调函数) / fs.appendFileSync(path,data)

删除文件

fs.unlink(path,回调函数) / fs.unlinkSync(path)

重命名文件

fs.rename(path,回调函数) / fs.renameSync(path)

检测文件是否存在

fs.existsSync(path)

拷贝文件

copyFileSync(path,path2) / copyFile(path,path2,回调函数)

文件流（E:\202008\NodeJS\2020-09-19\fs\stream\stream.js）

createReadStream创建可读取的流。

createWriteStream创建可写入的流。

pipe()用于将读取的流添加到写入流。

监听事件：

一旦触发事件会自动调用回调函数，

on(事件名称,回调函数)

HTTP协议。

是浏览器和web服务器之间的通信协议。

1、通用的头信息

request URL:请求的URL

request Method: get、post

status Code：响应状态码

1\*\* 正在响应。还未结束。

2\*\* 成功的响应。

3\*\* 响应的重定向，跳转。

4\*\* 客户端请求错误。

5\*\* 服务器错误。

2、响应的头信息resopnse

Content-Type 响应的内容类型 text/html

Location:响应后跳转的URL，结合状态码302使用。

3、请求的头信息request

4、请求体。

请求过程中表单传递的数据。

HTTP模块

创建web服务器，给浏览器提供服务。

createServe：创建web服务器

listen：给服务器设置端口

给服务器添加时间，用于接收请求以及作出响应

on('request',(request,response)=>{

request:请求的对象

request.url:请求的url

request.method请求的方法

response:响应的对象

res.writeHead() 设置响应的状态码和头信息

response.write() 设置响应的内容

resopnse.end() 结束并发送响应

})

创建web服务器，设置端口，根据浏览器请求的url做出响应，

/index 响应<h3>首页</h3>

/list 响应文件1.html 先同步读取文件数据，把读取结果响应

/study 跳转到 http://www.tmooc.cn

其他 响应404 not found