nodejs内置模块的使用实例

# 1.url模块

## 1.parse方法，把一个url字符串解析为一个对象

|  |
| --- |
| let url = require('url')  //1.parse方法，把一个网址解析为一个有很多属性的对象  let urlStr="http://www.nodejs.org/some/url/?with=query&param=that#about"  let ret = url.parse(urlStr,true)  console.log(ret); |
|  |

## 2、format方法

|  |
| --- |
| let url = require('url')  //1.parse方法，把一个网址解析为一个有很多属性的对象  let urlStr="http://www.nodejs.org/some/url/?with=query&param=that#about"  let ret = url.parse(urlStr,true)  // console.log(ret);  //2.//2 format方法，和上面的方法刚好反过来  let urlObj={      protocol:'https',      slashes:true,      host:"www.restelrico.com",      port:null,      hash:'#good',      search:'?recommand=de-la-casa&price=3700'  }  let urllink = url.format(urlObj)  console.log(urllink);//https://www.restelrico.com?recommand=de-la-casa&price=3700#good |

## 3.resolve方法，有点像url拼接，但是如果base里面有路径，这个路径被覆盖

|  |
| --- |
| let url1 = url.resolve('https://www.httpbin.ort','get')  // console.log(url1);//https://www.httpbin.ort/get  let url2 = url.resolve('https://www.httpbin.org/get','post')  console.log(url2);//https://www.httpbin.org/post 原来的网址有路径，直接覆盖 |

## 4.URLSearchPatams对象，以url.parse方法返回对象的search属性作为参数

|  |
| --- |
| //4.URLSearchParams  let link = "https://www.sogou.com/web?query=%E5%A5%B3%E5%B8%9D&\_asf=www.sogou.com"  let urlParams = new URLSearchParams(url.parse(link,true).search)  // console.log(urlParams);//URLSearchParams { 'query' => '女帝', '\_asf' => 'www.sogou.com' }  let urlString = 'https://www.baidu.com:443/path/index.html?id=2#tag=3'  let urlParams2 = new URLSearchParams(url.parse(urlString,true).search) //其实是一个Map数据结构  // console.log(urlParams2); //URLSearchParams { 'id' => '2' }  let urlString2 = "https://www.bing.com/images/search?q=car&qs=n&form=QBIDMH&first=1"  let urlParams3 = new URLSearchParams(url.parse(urlString2,true).search)  console.log(urlParams3); //URLSearchParams { 'q' => 'car', 'qs' => 'n', 'form' => 'QBIDMH', 'first' => '1' }  // console.log([...urlParams3]);  for(let [k,v] of urlParams3){       console.log(`key:${k},value:${v}`);  } |

## 5.URL类的searchParam

|  |
| --- |
| //5.URL类,它也一个searchParams属性也是可以解析search字符串  let myUrl = new URL('https://www.baidu.com:443/path/index.html?id=2#tag=3')  console.log(myUrl.searchParams); //URLSearchParams { 'id' => '2' }  console.log(myUrl.searchParams.get('id'));//2 |

# 2.querystring内置模块

## 1.parse方法，返回一个对象

|  |
| --- |
| let querystring = require('querystring')  let query = 'id=2&name=tongyi&from=北京'  let qret = querystring.parse(query)  console.log(qret);//[Object: null prototype] { id: '2', name: 'tongyi', from: '北京' }  console.log(qret.id); //2 |

## 2.escape方法，对汉字和非字符符号进行编码

|  |
| --- |
| let querystring = require('querystring')  //1.parse方法  let query = 'id=2&name=tongyi&from=北京'  let qret = querystring.parse(query)  // console.log(qret);//[Object: null prototype] { id: '2', name: 'tongyi', from: '北京' }  // console.log(qret.id); //2  //2.escape方法，对汉字和非字符符号进行编码  let escape = querystring.escape(query)  console.log(escape); //id%3D2%26name%3Dtongyi%26from%3D%E5%8C%97%E4%BA%AC |

## 3.url解码方法unescape

|  |
| --- |
| let querystring = require('querystring')  //1.parse方法  let query = 'id=2&name=tongyi&from=北京'  let qret = querystring.parse(query)  // console.log(qret);//[Object: null prototype] { id: '2', name: 'tongyi', from: '北京' }  // console.log(qret.id); //2  //2.escape方法，对汉字和非字符符号进行编码  let escape = querystring.escape(query)  // console.log(escape); //id%3D2%26name%3Dtongyi%26from%3D%E5%8C%97%E4%BA%AC  //3.url解码方法unescape  **let unescaped = querystring.unescape(escape)**  **console.log(unescaped); //id=2&name=tongyi&from=北京** |

## 4.stringfy方法添加分隔符并且对汉字编码

|  |
| --- |
| let querystring = require('querystring')  //1.parse方法  let query = 'id=2&name=tongyi&from=北京'  let qret = querystring.parse(query)  // console.log(qret);//[Object: null prototype] { id: '2', name: 'tongyi', from: '北京' }  // console.log(qret.id); //2  //2.escape方法，对汉字和非字符符号进行编码  let escape = querystring.escape(query)  // console.log(escape); //id%3D2%26name%3Dtongyi%26from%3D%E5%8C%97%E4%BA%AC  //3.url解码方法unescape  let unescaped = querystring.unescape(escape)  // console.log(unescaped); //id=2&name=tongyi&from=北京  //4.stringfy方法添加分隔符并且对汉字编码  let urlObj = {'id':2,name:'tongyi',from:'北京'}  let str\_url =querystring.stringify(urlObj,'/')  console.log(str\_url) //id=2/name=tongyi/from=%E5%8C%97%E4%BA%AC |

## 5. stringify方法默认会把中文和其他特殊字符进行编码，如果不想它进行编码就需要怎么写

|  |
| --- |
| let querystring = require('querystring')  let urlObj = {'id':2,name:'tongyi',from:'北京'}  //5.stringfy的特殊用法  let queryObj = querystring.stringify(urlObj,null,null,{      encodeURIComponent(string){          return querystring.unescape(string)      }  })  const { log } = require('console') //把控制台对象的log方法解构出来直接使用  log(queryObj) //id=2&name=tongyi&from=北京 |

# 3.http和https

## 1.http创建server，然后在里面显示用https获取一个网站的信息：

### http-https-get.js

|  |
| --- |
| let http = require('http')  let https = require('https')  let querystring = require('querystring')  //端口  let port = 2000  let server = http.createServer((req,res)=>{      let data = ''      // let testUrl ='https://www.google.com/search?q=https+%E6%B5%8B%E8%AF%95%E6%8E%A5%E5%8F%A3&sca\_esv=54af7326df9a1a15&sxsrf=ACQVn08iW2Fal8FPobZOZxqehJKruk4hFA%3A1708215311429&source=hp&ei=D0zRZaftF7-ywt0Pp\_GskA4&iflsig=ANes7DEAAAAAZdFaHwxRuf59qmsnXjlXK97d-6XqV1Ue&oq=https&gs\_lp=Egdnd3Mtd2l6IgVodHRwcyoCCAAyBBAjGCcyChAjGIAEGIoFGCcyBBAjGCcyChAAGIAEGIoFGEMyChAAGIAEGIoFGEMyBRAAGIAEMgUQABiABDIFEAAYgAQyBRAAGIAEMgUQABiABEjFH1AAWL4McAB4AJABAJgBxgKgAasIqgEHMC4zLjEuMbgBAcgBAPgBAcICCxAuGIAEGMcBGNEDwgIFEC4YgAQ&sclient=gws-wiz'      // let testUrl ='https://www.google.com/search?q=https' //ok      // let testUrl ='https://en.wikipedia.org/wiki/Girls\_(TV\_series)' //ok      // let testUrl ='https://dictionary.cambridge.org/zhs/%E8%AF%8D%E5%85%B8/%E8%8B%B1%E8%AF%AD-%E6%B1%89%E8%AF%AD-%E7%AE%80%E4%BD%93/pussy' //ok      // let testUrl ='https://sogou.com/web?query=girls&\_asf=www.sogou.com&\_ast=1708216463&w=01019900&p=40040100&ie=utf8&from=index-nologin&s\_from=index&sut=2532&sst0=1708216463282&lkt=5%2C1708216460751%2C1708216461555&sugsuv=1708216456647776&sugtime=1708216463282'      // let testUrl2 ='https://reqable.com/zh-CN/docs/rest/'      let testUrl2 ='https://github.com/kennycaiguo'      https.get(testUrl2,(result)=>{          result.on('data',(chunk)=>{              data += chunk          })          result.on('end',()=>{              res.writeHead(200,{                  "Conten-type":"application/json;charset=utf-8",                  "access-control-allow-origin":"\*"              })              res.write(data)             res.end()          })      })  })  server.listen(port,()=>{      console.log(`server is ready at:http://localhost:${port}/`);  }) |

2.