

```
1 http://www.puppeteerjs.com/
```

```
1 let a = new URL('https://www.baidu.com') //解析url地址
```

使用axios

```
1 npm install axios
```



```
patch: [Function: wrap],
defaults: {
  adapter: [Function: httpAdapter],
  transformRequest: [ [Function: transformRequest] ],
  transformResponse: [ [Function: transformResponse] ],
  timeout: 0,
  xsrfCookieName: 'XSRF-TOKEN',
  xsrfHeaderName: 'X-XSRF-TOKEN',
  maxContentLength: -1,
  validateStatus: [Function: validateStatus],
  headers: {
    common: [Object],
    delete: {},
    get: {},
    head: {},
    post: [Object],
    put: [Object],
    patch: [Object]
  }
},
interceptors: {
  request: InterceptorManager { handlers: [] },
```

输出axios可以看到很多方法

最基本的爬虫

```
1 let a = 'https://www.qweather.com/business/map'
2 const axios = require('axios')
3 //或者request.get 首先引入request
4 axios.get(a)
5 .then(res=>{
6     console.log(res);
7 })
8
9 let httpURL = 'https://www.baidu.com'
10 axios.get(httpURL,{
11     headers:{'X-Requested-With':"XMLHttpRequest"},
12 }).then(function(res) {
13     console.log(res)
14 })
```

浏览器打开网页时做了什么我们也要做什么 伪装成浏览器

爬取电影网站

- //获取起始页面所有分类
- //获取分类里的电影链接
- //根据电影链接获取电影的详细信息

1 正则表达式爬取需要的内容

\$	匹配输入字符串的结尾位置。如果设置了 RegExp 对象的 Multiline 属性，则也匹配 '\n' 或 '\r'。要匹配 \$ 字符本身，请使用 \\$。
()	标记一个子表达式的开始和结束位置。子表达式可以获取供以后使用。要匹配这些字符，请使用 \(和 \)。
*	匹配前面的子表达式零次或多次。要匹配 * 字符，请使用 *。
+	匹配前面的子表达式一次或多次。要匹配 + 字符，请使用 \+。
.	匹配除换行符 \n 之外的任何单字符。要匹配 .，请使用 \.。
[标记一个中括号表达式的开始。要匹配 [，请使用 \[。
?	匹配前面的子表达式零次或一次，或指明一个非贪婪限定符。要匹配 ?，请使用 \?。
\	将下一个字符标记为或特殊字符、或原义字符、或向后引用、或转义。在类

	符。例如， 'n' 匹配字符 'n'。'\n' 匹配换行符。序列 '\\' 匹配配 "("。
^	匹配输入字符串的开始位置，除非在方括号表达式中使用，当该表达式中使用时，表示不接受该方括号表达式中的字符集合。要自身，请使用 \^。
{	标记限定符表达式的开始。要匹配 {，请使用 \{。
	指明两项之间的一个选择。要匹配 ，请使用 \ 。

```

1 let reg = /<span></span>/
2 /let result = reg.exec(body)
3 //exec匹配内容 需要循环爬取
4 let reg = /<a href=".*?">.*?</a>/igs
5     let arr = []
6     let result
7     while( result = reg.exec(res.data) )
8     {
9         arr.push(result[0])
10         //
11     }
12     console.log(arr)
13
14     var str="<p>1213214312441</p><p>1213214312441</p><p>1我是你巴巴</p><p>sb41</p><p>1213214312441</p>"
15     document.write(str.match(/<p>(.*?)</p>/g))
16

```

实验表明 match更好用

爬取时要注意 () .等需要转义

cheerio插件（以jquery为核心）

```

1 let cheerio = require("cheerio")
2 //axios获取res
3 let $ = cheerio.load(res.data)
4 axios.get(httpUrl).then(res=>{
5     let $ = cheerio.load(res.data)

```

```

6      $(".col-sm-9 a").each((i,ele) =>{
7          console.log($(ele).attr("href"))          //注意给ele加上$
8          let title = ....find(".random_title").text()
9      })
10 })

```

爬取表情包 图片的地址需要 http -> http

```

1
2  async function  parsePage(url) {
3      let res = await axios.get(url)
4      let $ = cheerio.load(res.data)
5      $(".artile_des table tbody tr td a img").each((i,ele) =>{
6          let mesUrl = $(ele).attr("src")
7          let newUrl = mesUrl.substr(0,4) + 's' + mesUrl.substr(4,mesUrl.length - 4 + 1)
8          console.log(newUrl);
9      })
10 }

```

得到结果 放到本地文件

```

1  let ws = fs.createWriteStream('img/ / ')
2  axios.get(imgUrl,{responseType:'stream'}).then(function(){
3      res.data.pipe(ws)//导入到写入流
4
5  })

```

```

1  axios.get后面可以直接.then
2  /(.*?)\d/igs //匹配文本

```

反爬虫

```
res = await axios.get(httpUrl,{
  proxy: {
    host: '127.0.0.1',
    port: 9000,
    auth: {
      username: 'mikeymike',
      password: 'rapunz31'
    }
  },
})
```

可以为axios设置代理

```
1 async function download(url,title) {
2   let res = await axios.get(url,{responseType:"stream"})
3   let wx = fs.createWriteStream('./mp3/' + 'title' + '.mp3')
4   res.data.pipe(ws)
5   res.data.on('close',function() {
6     ws.close()
7   })
8 }
```

追加

```
1 fs.writeFile("music.txt",content,{flag:'a'},function() {
2   console.log("写入完成")
3 })
```

Puppeteer

无头浏览器

倾向于抓取SPA

```
1 let puppeteer = require('puppeteer')
2 async function test() {
3   let option = {
4     defaultViewport:{
5       width:1400,
6       height:800
7     }, //默认视图窗口
8     headless:false,
9     slowMo:1000, //放慢毫秒数
10  }
11  let browser = await puppeteer.launch(option) //可以配置为无界面浏览器 性能更高
12  let page = await browser.newPage()
13  await page.goto("https://www.baidu.com")
14  await page.screenshot({path:"screenshot.png"})
15  page.$$('selector') //获取选择的内容
16  await page.$$eval("#menu li a", (elements) => {
17    elements.forEach((item, i) => {
18      console.log(item.innerHTML)
19    })
20  })
21  page.on('console', function(...arg) {
22    console.log(args)
23  })
24 }
25 test()
```

对Dom元素进行操作

```
1 elementHandles = await page.$$("menu li a")
2 elementHandles[2].click()
3 inputEle = await page.$(".search1 .formhue")
4 await inputEle.focus()
5 await page.keyboard.type("小丑") //输入小丑
6 let btnEle = await page.$(".....")
7 await btnEle.click()
```

page.waitForSelector(