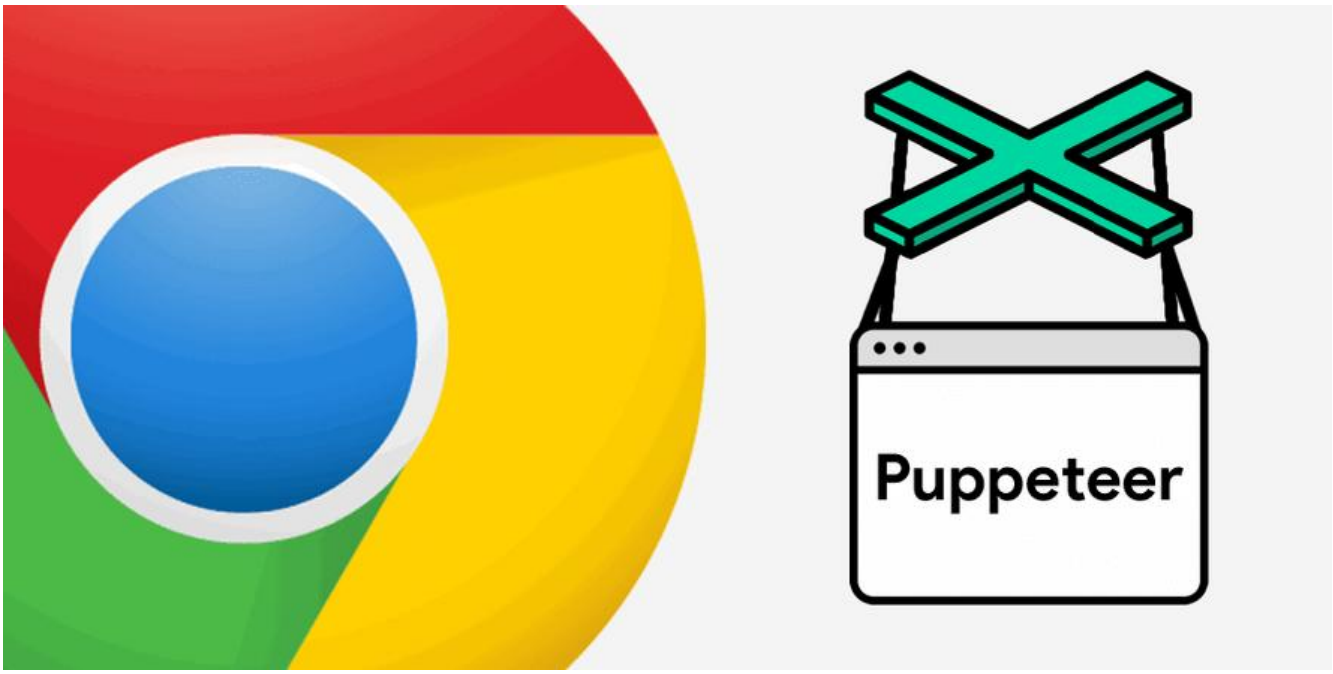
链滴

Puppeteer 爬取豆瓣小组公开信息

作者：martinageradams

原文链接：https://ld246.com/article/1590201669356

来源网站：链滴



题外话

老王，最近取了笔名。不仅仅是笔名，字、号也统统安排。

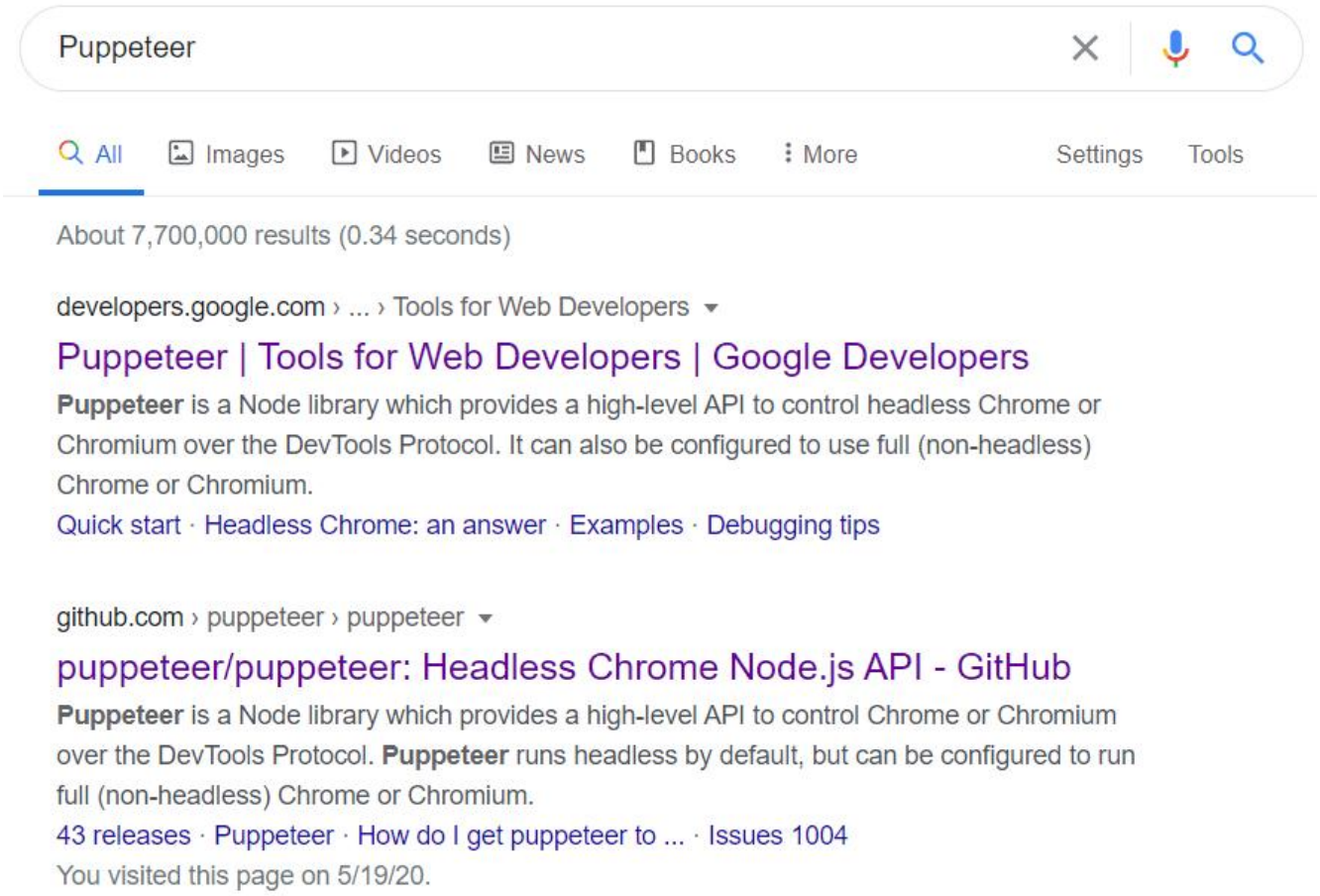
上官追风，字追风，号追风居士。

非要给它一个解释的话，那就是「追风少年宅家里」。

老王的行文路线其实就是他的思维路线路。

Puppeteer

面对未知的事物，最好的老师显然是搜索引擎，而搜索引擎中公认最好的又是 Google 搜索。



Puppeteer 文档

Github: github.com/puppeteer/puppeteer

英文文档：pptr.dev

中文文档：zhaoqize.github.io/puppeteer-api-z...

Puppeteer 简介

以下介绍摘录自中文文档。

Puppeteer 读作 /puh · puh · teer/，是一个 Node 库，它提供了一个高级 API 来通过 DevTools 议控制 Chromium 或 Chrome。Puppeteer 默认以 headless 模式运行，但是可以通过修改配置文运行 “有头” 模式。

- 生成页面 PDF。
- 抓取 SPA「单页应用」并生成预渲染内容（即 SSR「服务器端渲染」）。
- 自动提交表单，进行 UI 测试，键盘输入等。
- 创建一个时时更新的自动化测试环境。使用最新的 JavaScript 和浏览器功能直接在最新版本的 Chrme 中执行测试。
- 捕获网站的 timeline trace，用来帮助分析性能问题。
- 测试浏览器扩展。

原文链接：Puppeteer 爬取豆瓣小组公开信息

项目背景

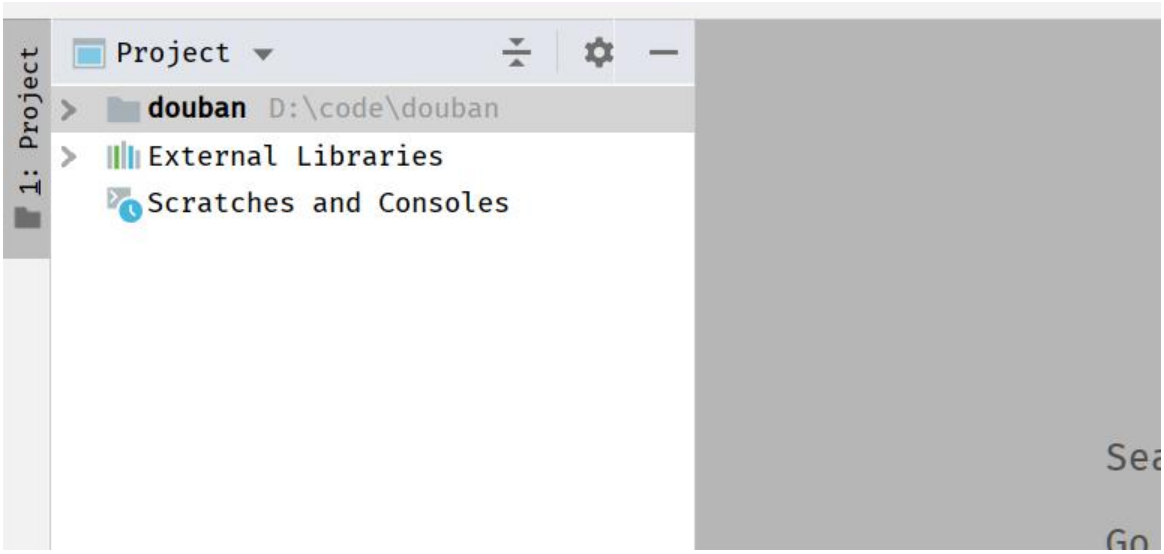
老王开始了电鸭社区「征稿」板块的事务，需要大量联系征稿人来电鸭社区发征稿贴。

偶然发现一个豆瓣小组有征稿人签到帖，那怎么办？手动复制粘贴是不可能的，马上动手写小爬虫。

代码实战

第一步：创建项目

- 创建一个目录 douban



原文链接：Puppeteer 爬取豆瓣小组公开信息

- 创建 douban.js 文件
- 粘贴官网的示例代码

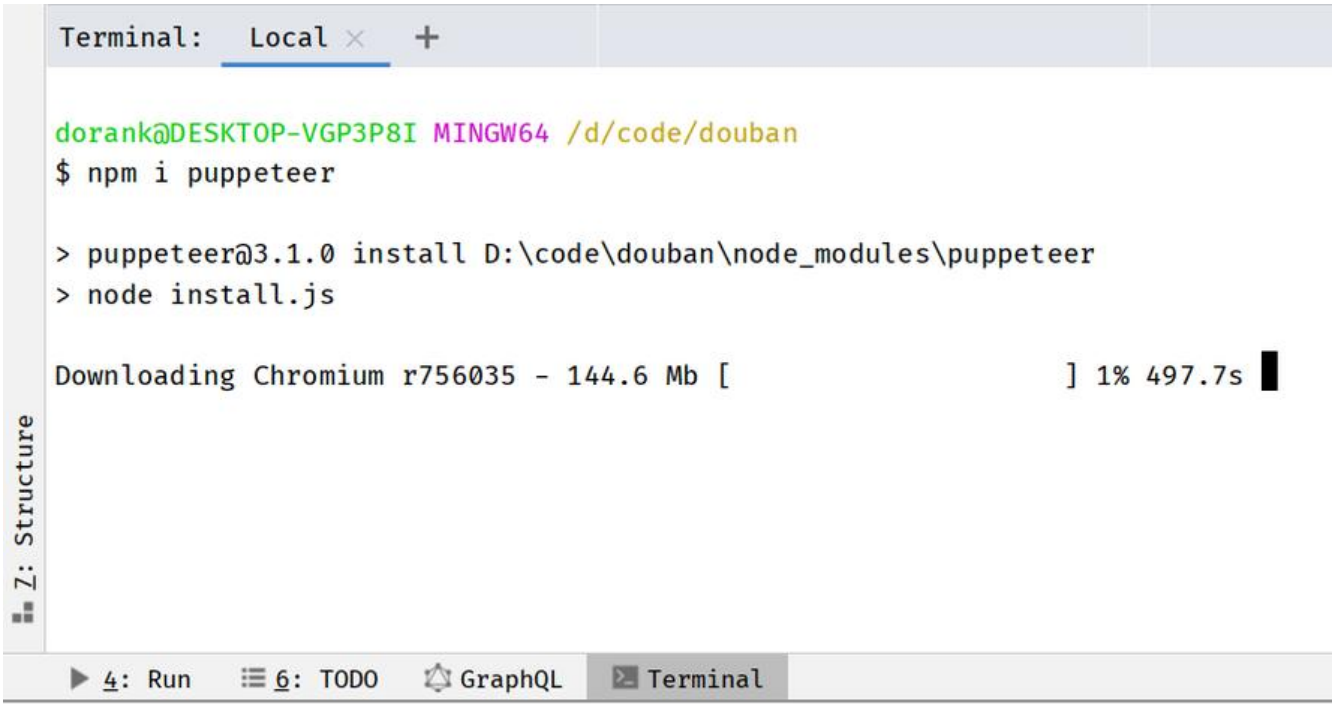
```
const puppeteer = require('puppeteer');

(async () => {
    const browser = await puppeteer.launch();
    const page = await browser.newPage();
    await page.goto('https://douban.com');
    await page.screenshot({path: 'example.png'});
    await browser.close();
})();
```

- npm 安装 Puppeteer

别急，还不能运行代码呢。开启终端到项目根目录 npm 安装 Puppeteer

npm i puppeteer

需要等待 Chromium 安装完，网络不好的小伙伴，自己想想办吧。



- 修改 package.json 文件

```
{
  "name": "douban",
  "version": "1.0.0",
  "scripts": {
    "start": "node ./douban.js"
  },
  "dependencies": {
    "puppeteer": "^3.1.0"
  }
}
```

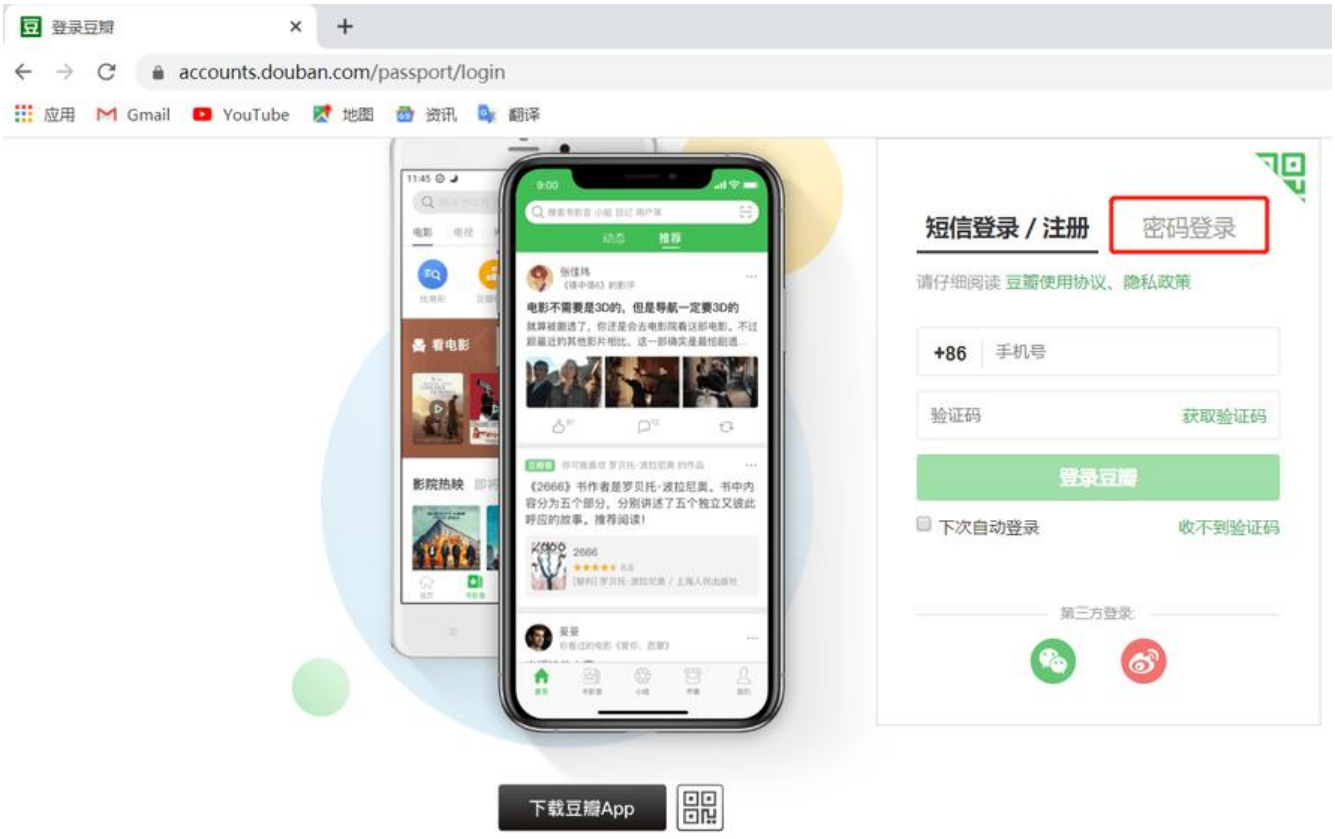
原文链接：Puppeteer 爬取豆瓣小组公开信息

第二步：模拟登陆

访问目标页面，发现需要登陆。

- 分析登陆页面结构

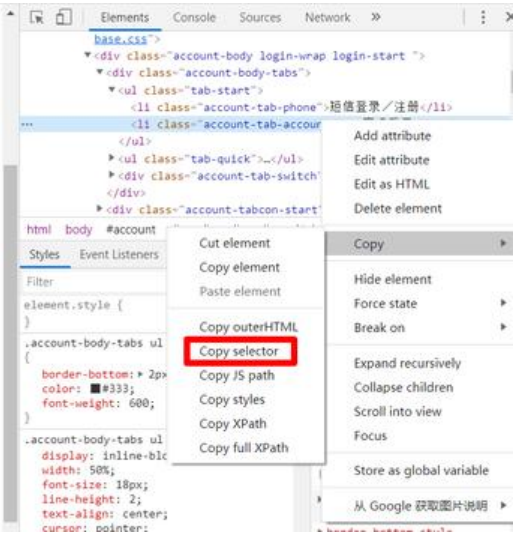
我选择了密码登录，降低复杂度。



原文链接：Puppeteer 爬取豆瓣小组公开信息

我们需要干什么呢?

- 打开页面
- 点击密码登录
- 输入账号
- 输入密码
- 点击登陆



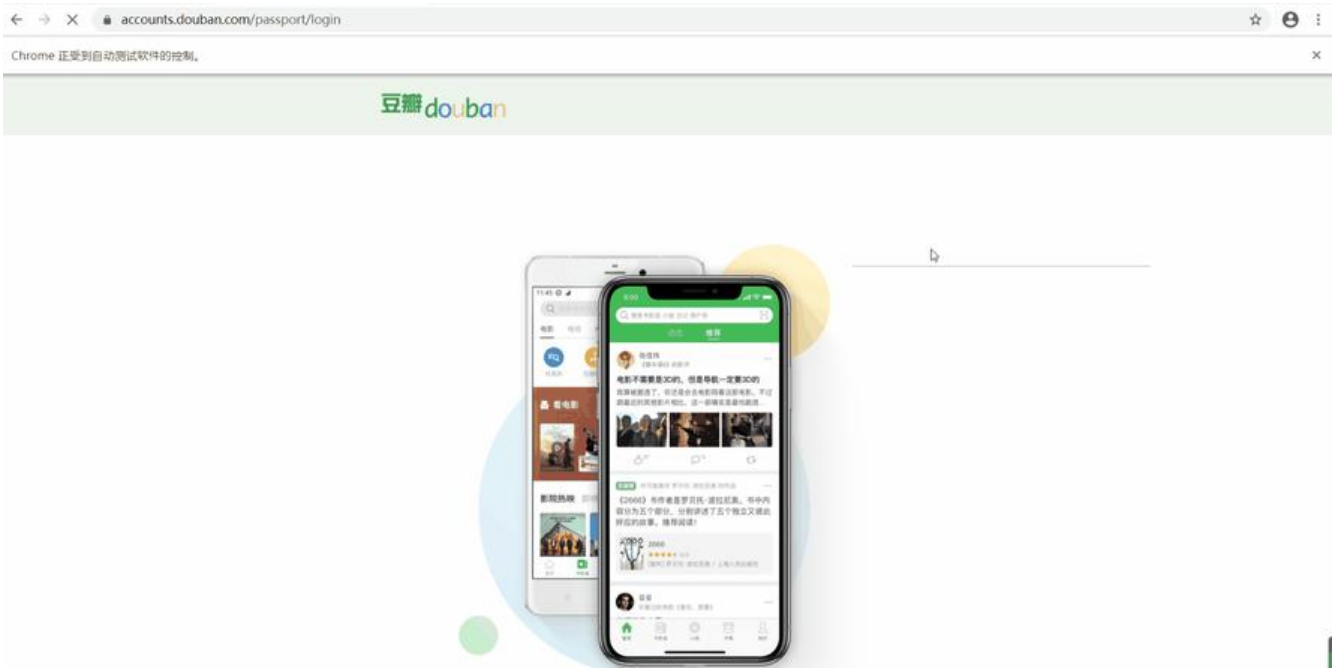
- 代码示例

```
const puppeteer = require('puppeteer');

(async () => {
    const browser = await puppeteer.launch({
        headless: true,
        timeout: 50000
    })

    const page = await browser.newPage()

    // 去豆瓣那个页面
    await page.goto('https://accounts.douban.com/passport/login', {
        waitUntil: 'networkidle2'  // 网络空闲说明已加载完毕
    });

    // 点击搜索框拟人输入
    const clickPhoneLogin = await page.$('.account-tab-account')

    await clickPhoneLogin.click()

    const name = 'xxxxxxxxxx'
    await page.type('input[id="username"]', name, {delay: 0})

    const pwd = 'xxxxxxxxxx'
    await page.type('input[id="password"]', pwd, {delay: 1})
```

原文链接：Puppeteer 爬取豆瓣小组公开信息

```
    // 获取登录按钮元素
    const loginElement = await page.$('div.account-form-field-submit > a')

    await loginElement.click()

    await page.waitForNavigation()

    await browser.close()
})();
```

最终效果



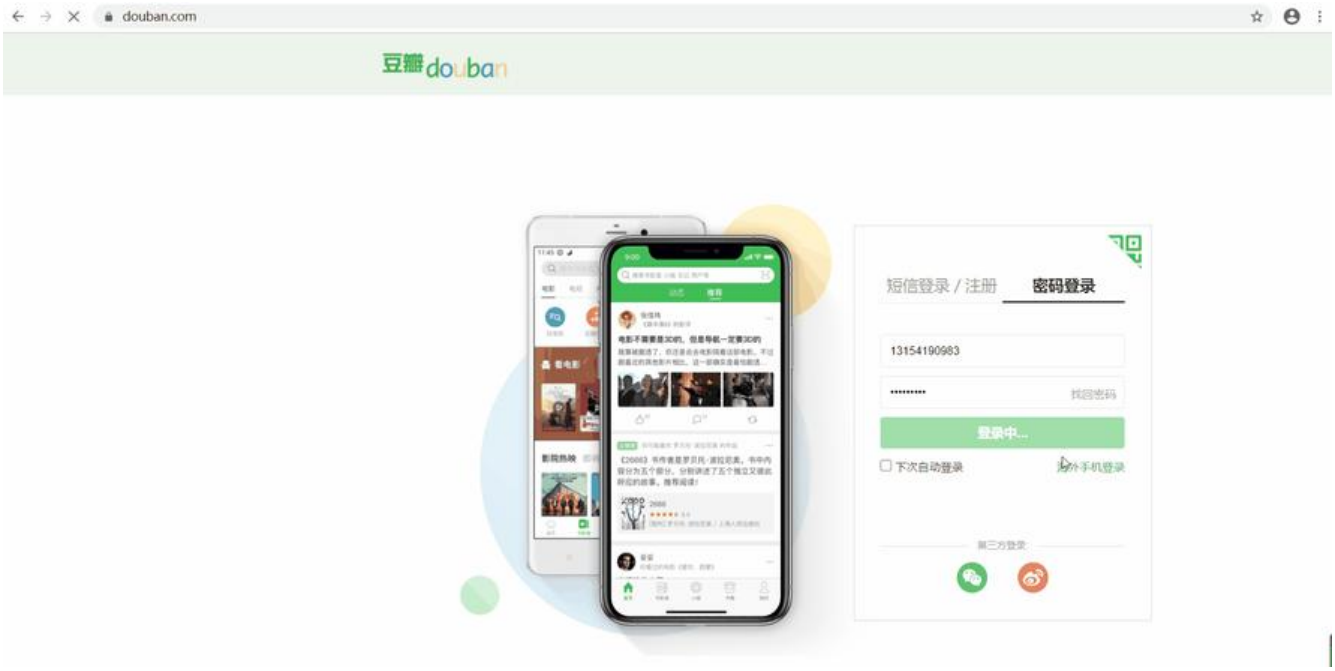
第三步：爬取数据

有了前面的基础，后面我就不详细讲啦。

```
const puppeteer = require('puppeteer');

(async () => {
    const browser = await puppeteer.launch({
        headless: false,
        timeout: 50000
    })

    const page = await browser.newPage()

    // 去豆瓣登陆页面
    await page.goto('https://accounts.douban.com/passport/login', {
        waitUntil: 'networkidle2'  // 网络空闲说明已加载完毕
    });

    // 点击搜索框拟人输入
    const clickPhoneLogin = await page.$('.account-tab-account')
```

```
    await clickPhoneLogin.click()

    const name = 'xxxxxxxx'
    await page.type('input[id="username"]', name, {delay: 0})

    const pwd = 'xxxxxxxx'
    await page.type('input[id="password"]', pwd, {delay: 1})

    // 获取登录按钮元素
    const loginElement = await page.$('div.account-form-field-submit > a')

    // 点击按钮，开始登陆
    await loginElement.click()

    await page.waitForNavigation()


    // 目标页面 url
    let url = 'https://www.douban.com/group/topic/112565224/?start='
    // 翻页参数
    let pages = [0, 100, 200, 300, 400, 500]

    // 定义爬取函数
    async function next(url) {
        await page.goto(url, {
            waitUntil: 'networkidle2'  // 网络空闲说明已加载完毕
        })

        return await page.$$eval("div.reply-doc.content > p", e => {

            let a = []

            e.forEach(element => {
                a.push(element.innerText)
            })

            return a
        })
    }

    // 拼接文本字符串
    let data = ''

    for (const index of pages) {
        let res = await next(url + index)
        data = res.join('\n\n\n----------------------------------------------------------\n\n') + data
    }

    // 查看一下数据
    console.log(data)

    await browser.close()
})();
```

最终效果



第四步：写入数据

```
const puppeteer = require('puppeteer');
const fs = require('fs');

(async () => {
    const browser = await puppeteer.launch({
        headless: false,
        timeout: 50000
    })

    const page = await browser.newPage()

    page.setViewport({
        width: 1920,
        height: 1080
    })

    // 去豆瓣登陆页面
    await page.goto('https://accounts.douban.com/passport/login', {
        waitUntil: 'networkidle2'  // 网络空闲说明已加载完毕
    });

    // 点击搜索框拟人输入
    const clickPhoneLogin = await page.$('.account-tab-account')

    await clickPhoneLogin.click()

    const name = 'xxxxxxxx'
    await page.type('input[id="username"]', name, {delay: 0})

    const pwd = 'xxxxxxxx'
```

```
    await page.type('input[id="password"]', pwd, {delay: 1})

    // 获取登录按钮元素
    const loginElement = await page.$('div.account-form-field-submit > a')

    // 点击按钮，开始登陆
    await loginElement.click()

    await page.waitForNavigation()


    // 目标页面 url
    let url = 'https://www.douban.com/group/topic/112565224/?start='
    // 翻页参数
    let pages = [0, 100, 200, 300, 400, 500]

    // 定义爬取函数
    async function next(url) {
        await page.goto(url, {
            waitUntil: 'networkidle2'  // 网络空闲说明已加载完毕
        })

        return await page.$$eval("div.reply-doc.content > p", e => {

            let a = []

            e.forEach(element => {
                a.push(element.innerText)
            })

            return a
        })
    }


    // 拼接文本字符串
    let data = ''

    for (const index of pages) {
        let res = await next(url + index)
        data = res.join('\n\n\n------------------------------------------------------------\n\n') + data
    }

    // 写入文件
    fs.writeFile('douban.txt',data,'utf8',function(error){
        if(error){
            console.log(error);
            return false;
        }
        console.log('写入成功');
    })

    await browser.close()
})();
```

实战反思

- 代码还需要优化，尤其是翻页写的很差。
- 能不能分模块来实现。这段代码中，模拟登陆、爬取目标、写入文件都是揉在一起的。
- 暂时就这些啦。

完整代码

gist.github.com/w3cfed/75217423f86...

```
const puppeteer = require('puppeteer');
const fs = require('fs');

(async () => {
    const browser = await puppeteer.launch({
        headless: false,
        timeout: 50000
    })

    const page = await browser.newPage()

    // 去豆瓣登陆页面
    await page.goto('https://accounts.douban.com/passport/login', {
        waitUntil: 'networkidle2'  // 网络空闲说明已加载完毕
    });

    // 点击搜索框拟人输入
    const clickPhoneLogin = await page.$('.account-tab-account')

    await clickPhoneLogin.click()

    const name = 'xxxxxxx'
    await page.type('input[id="username"]', name, {delay: 0})

    const pwd = 'xxxxxxxx'
    await page.type('input[id="password"]', pwd, {delay: 1})

    // 获取登录按钮元素
    const loginElement = await page.$('div.account-form-field-submit > a')

    // 点击按钮，开始登陆
    await loginElement.click()

    await page.waitForNavigation()


    // 目标页面 url
    let url = 'https://www.douban.com/group/topic/112565224/?start='
    // 翻页参数
    let pages = [0, 100, 200, 300, 400, 500]

    // 定义爬取函数
```

```
    async function next(url) {
        await page.goto(url, {
            waitUntil: 'networkidle2'  // 网络空闲说明已加载完毕
        })

        return await page.$$eval("div.reply-doc.content > p", e => {

            let a = []

            e.forEach(element => {
                a.push(element.innerText)
            })

            return a
        })
    }


    // 拼接文本字符串
    let data = ''

    for (const index of pages) {
        let res = await next(url + index)
        data = res.join('\n\n\n----------------------------------------------------------------\n\n') + data
    }

    // 写入文件
    fs.writeFile('douban.txt',data,'utf8',function(error){
        if(error){
            console.log(error);
            return false;
        }
        console.log('写入成功');
    })

    await browser.close()
})();
```