数据采集的另一种思路 - 浏览器脚本注入

作者：maixiaojie
原文链接：https://ld246.com/article/1559269309863
来源网站：链滴

昨天想去极客时间把购买的一个专栏里的数据扒下来，发现之前写的python脚本不能用了，原因是们网站做了限流、也加了http时间戳的一些校验。我们可以将之前的python脚本进行改进，用ip代理来处理限流，寻找时间戳验证的规则就可以解决。

但是这次我们用了另外的一种爬虫的思路，就是我们直接写一些js脚本，在对方的网站里运行，去请相应的接口，从而得到想要的数据。

这种思路其实见过很多例子，之前有一个很火的，qq空间自动点赞的脚本，看过它的源码，其实很简，就是直接去操作dom，然后触发一些事件。

另外一个很火的例子，github上很火的一个repo， fuckZhihu，据说是winter当年退知乎时写的，自己在知乎的数据保存下来。

下面是这次实践的内容：

获取文章id集合

刚进入专栏的时候会有一个获取左侧文章列表集合的请求，在这个接口里，我们就能获取到当前专栏所有请求。

```
/**
 * 运行在浏览器控制台中
 * @type {Array}
 */

// 要爬取的文章id集合  geek网站做了限流，所以分两次请求。当然你也可以用代理什么的
var ids = [77345, 77749, 77804, 78158, 78168, 78884, 79319, 79539, 80011, 80021, 80042, 80240, 80260, 80311, 81730, 82
var ids2 = [86400, 86823, 87179, 87234, 87808, 88275, 88538, 88827, 89151, 89491, 89832, 90148, 90485, 90998, 91325, 9
```

这个专栏大概有50多篇文章，因为限流的原因，我们分成两次进行请求。

注入FileSaver.js

FileSaver是一个运行在浏览器中，将数据下载为json或者excel文件的库。

```
/**
 * 初始化    文档中注入FileSaver.js 并执行请求开始的方法
 * @return {[type]} [description]
 */
function init() {
    var src = 'https://cdn.bootcss.com/FileSaver.js/2014-11-29/FileSaver.js';
    var script = document.createElement('script');
    script.src = src;
    var heads = document.getElementsByTagName("head");
    if (heads.length)
        heads[0].appendChild(script);
    else
        document.documentElement.appendChild(script);
    script.onload = function() {
        console.log('script loaded')
        start()
    }
}
```

我们在这里创建一个script标签，并将这个标签插入到文档中。

```
// 将数据转存为json文件
function downloadJson(data) {
    var blob = new Blob([JSON.stringify(data)], { type: "" });
    saveAs(blob, "data.json");
}
```

我在这里写了一个方法downloadJson，我们将等会获取到的数据传到这里来，就可以下载这个json件了。

创建请求

创建ajax请求，请求文章详情的接口。

```
// 爬取一篇文章 就是一个ajax请求
function fetch(id) {
    var data = JSON.stringify({
        "include_neighbors": "false",
        "id": id
    });
    var xhr = new XMLHttpRequest();
    xhr.withCredentials = true;
    xhr.addEventListener("readystatechange", function() {
        if (this.readyState === 4) {
            var res = JSON.parse(this.responseText);
        }
    });

    xhr.open("POST", "https://time.geekbang.org/serv/v1/article");
    xhr.setRequestHeader("content-type", "application/json");
    xhr.send(data);
}
```

这里我们用原生的js来写的，是一个post请求，res就是我们得到这个接口的返回值，我们将需要的数从这个返回值中取出来就可以了。

上面说的是单个请求的实现。多个请求的实现如下图所示。

```
function start() {
    // 依次请求
    for (var i = 0; i < ids2.length; i++) {
        (function(i) {
            setInterval(fetch(ids2[i]), 3000)
        })(i)
    }
}
```

然后我们将数据保存一下：

```
var res = JSON.parse(this.responseText);
if (res.code == 0) {
    var data = res.data;
    var item = {
        id: data.id,
        pid: data.cid,
        article_content: data.article_content,
        article_cover: data.article_cover,
        article_ctime: data.article_ctime,
        article_title: data.article_title,
        audio_download_url: data.audio_download_url,
        audio_size: data.audio_size,
        audio_time: data.audio_time,
        audio_url: data.audio_url
    }
    rs.push(item);
```

所有的结果都放在rs这个数组中了。

下载数据

我们将所有数据放在了一个数组中，在最后一次请求结束的时候，执行我们写好的downloadJson方进行下载就可以了。

```
// 如果当前是最后一个就下载文件
if (id == ids2[ids2.length - 1]) {
    downloadJson(rs)
}
```

导入数据库

json文件导入数据库网上有很多的工具，我这次是用之前写好的脚本。

这个脚本在我的github上面，是用nodejs写的，地址：tomysql.js

最后

我们这次没用通用的做法，模拟请求，或者模拟浏览器，而是直接利用浏览器来采集数据，当然也要据实际情况去选择用哪种做法。

完整的脚本： geek.js