Node.js 爬虫从 0 到 1

写在前面

我们经常会听说爬虫这个词语，但是却从来没有见过这个'虫子'，在我们日常生活中，每天使用的百度，谷歌，搜狗，360等搜索引擎的背后，都是无数的爬虫在支持，相信很多人都听过SEO，其实，SEO就是一门如何让爬虫更快更好的爬取到你的网站的技术，当然，如果你有钱，完全可以搞个竞价排名!

课程简介

其实很多语言都能写爬虫，最著名的应该是Python，它是一门强大的语言，建议有精力的人学一学，以后大数据，深度学习肯定是大方向!当然，除了Python还有很多语言能写爬虫，比如今天我们要讲的Node.js。

简单的介绍一下Node.js，它可以在服务端运行js，做过前端开发的程序员肯定很熟悉，js是一门弱语言，但现在发展很迅速，茁长成长，在Nodejs出现以后，使得js不光能写前端的动态效果，交互效果，还能写web服务器，我们甚至能用Nodejs去打包桌面端程序，从此，前端工程师的触角向后延伸了一大块。

对于前端工程师来说有时候可能想爬取点简单的页面，那么Nodejs将是我们的好帮手，当然了，闲来无事的时候，你也可以爬取一点福利站之类的，你懂得!

这节chat可能只会讲一点简单的爬虫，真正的爬虫与反爬虫博弈十分厉害，后期会推出一系列的教程来讲解，正在求职之中，时间紧，任务重，如果写的不好，还请提出，我会认真改正，谢谢!

- 下载地址：https://nodejs.org/en/download/（此网址为node官网）
- 这个是node中文网http://nodejs.cn/download/

Windows和Mac直接下载安装包，双击打开安装就好了。

下面重点来说一下linux系统下的安装，为什么要重点说Linux系统呢?因为我用的是Linux，哈哈，当然这个不是主要原因，主要原因是目前大多数服务器环境是Linux，如果我们将来将node的程序部署的服务器上，那这是很有用的。

Linux参照这个去安装，https://nodejs.org/en/download/package-manager/。这个网址有时候会上不了，你懂得，我在这贴出几个常用的服务器系统的安装方式。

安装完成后后，可以执行npde -v和npm -v确认是否安装成功。

```
Debian and Ubuntu
curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -
sudo apt-get install -y nodejs


RHEL,  CentOS or Fedora
curl --silent --location https://rpm.nodesource.com/setup_8.x |
sudo bash -
sudo yum -y install nodejs
```


爬虫简介

话不多说，直奔主题，下面来讲解一下爬虫是啥。

说了半天，爬虫是什么呢?其实爬虫就是你，说的具体点，当你坐在电脑前面打开这篇chat，你就充当了一个爬虫的角色，那我们来分解一下你打开这篇chat的行为:

> 输入固定的网址---->点击回车---->看到这篇chat

其实还能向下再去细分

```
请求---->解析&处理数据---->数据去重---->保存数据
```

今天我们以第二种模型来讲解，先让大家对爬虫有个初步的了解。

模型Node化

这节的标题有点奇怪，也不是太贴切，这节主要讲的是，对应上述模型，在node中我们使用那些对应的模块去操作呢?

请求	request模块
解析&数据处理	cheerio模块
保存数据	fs模块

为保证大多数人听懂（很多朋友是从python群里过来的），本次chat代码将不会使用promise，class，async & await等ES6，7的代码，尽可能的简单，但我希望大家能学习上面几个代码，如果后续推出高阶课程，我会使用他们来构建爬虫代码，解决node爬虫异步的问题.只要明白原理了，那js代码还不是你想怎么写就怎么写.说了这么多，开始写代码!

从0到1，开始代码

首先说一下，今天我们爬取的是花瓣美素网址http://www.meisupic.com/。里面有大量的图片供我们做设计使用.当我们拿到想要爬取的网址时，首先要分析这个网站的url。

```
http://www.meisupic.com/     这个网址相当与主站地址，下面的为分页地址
http://www.meisupic.com/topic.php?topic_id=5
http://www.meisupic.com/topic.php?topic_id=20
http://www.meisupic.com/topic.php?topic_id=1
```

打开美素网的页面，F12查看，我们会发现其实有多种方法爬取这个网站

```
5. mkdir data            // 新建data文件夹，用来存储csv文件，如果不新建，
后期会报错
5. npm install request --save       // 安装request模块
6. npm install cheerio --save       // 安装cheerio模块
7. fs模块是node自带的不需要安装
8. 将文件夹在vscode里面打开，开始编写代码
```

以下是正式代码：

```
const request = require('request');
const cheerio = require('cheerio');
const fs = require('fs');
// 上面三行代码是导入我们所需的三个模块
// 下面的代码，我将会用最简单的js代码书写，会显得比较low，请见谅！
/*

  http://www.meisupic.com/topic.php?topic_id=1
  http://www.meisupic.com/topic.php?topic_id=5
  http://www.meisupic.com/topic.php?topic_id=20
  分析上面的代码，可将代码分成两部分，
  http://www.meisupic.com/topic.php? 与 topic_id=1
  由于我们不知道这个网址具体有多少个子页面，所以我们可以去
  http://www.meisupic.com/topic.php上将他全部的子页面爬取下来
  再次进行抓取

*/
// 主站地址
let Url = 'http://www.meisupic.com/topic.php'
// request 负责请求
request({url : Url},  function (err,  res,  body) {
    if (err) {
        console.log(err)
    } else {
/*
   cheerio模块负责html的解析，有兴趣的可以直接在上面打印body，这里只需要把
   返回的body load进去就可以进行解析了.
```

```
/*
    此处做了简单的header伪装，但header伪装还有很多参数我没有写上，比如说有些网址是有防盗链的，那我们就要伪造一个referer来破解防盗链
*/
            request({
                url:reUrl,
                method: 'GET',
                headers: {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML,  like Gecko) Chrome/51.0.2704.106 Safari/537.36'}
            },  function (err,  response,  body) {
                let $ = cheerio.load(body)
                let pichref = $('.imgList .imgItem a img')
                let picname = $('.ui_cover dl')
                for(let j=0;j<pichref.length;j++){
                   let downpichref = pichref[j].attribs['data-original']
                   let downpicname = picname[j].attribs.title
                   save(downpicname,  downpichref)
/*  以下代码是调用下载程序的，不建议调用，会给网站造成很多流量浪费.
                   downloadImg(downpichref,  downpicname, function () {
                    console.log(downpicname + 'upload 完成');
                });
*/
                }
            })
        }
    }
})
/*
    下面的代码是使用fs去保存文本，我并没有真正的去下载这些图片，
    考虑到版权以及下载图片会增加花瓣网站的流量，
    我只是简单的将链接保存在了一个csv文件里
    还可以使用mongoose链接上mongoDB，存数据库里.
    那如果我们要下载要怎么写呢?可以用pipe通道，或者是使用专用的下载模块去下
```

```
                console.log('已成功爬取'+ k + '条')
                k = k + 1
            }

        });

    }
    /*
        此处为保存url的代码，只需要在上面的函数中调用就可以了
    */

    function downloadImg(url,  filename,  callback) {
        var stream = fs.createWriteStream('./images/' + filename);
        request({url:url}).on('error', function(){
            console.log('done no');
        }).pipe(stream).on('close',  callback);
    /*
        这段是下载function的代码
        有兴趣的可以自己研究
    */
    }


    /*
        代码没有使用es6，7的新特性，有兴趣的朋友可以自己修改代码，只要原理清楚就可以了
        后期会推出node爬虫更详细的教程，我会将此代码开源到github，大家一定去我的github看看
        记得给个start，别问为什么，你会懂的！
    */
```


HTTP请求头模拟

本次代码中的请求头模拟并没有涉及到太多，后续的课程里会详细的讲解，爬虫需要模

我已经将本文的代码开源到github了，会给大家提供三个版本的代码，一个是上面演示的代码，2.0版本的是第一种方式的代码，用了一些es6语法，第三个版本是让好基友帮忙写的class版本，建议大家参考1.0版本来熟悉原理，3.0版本要好好看，以后尽量写成3.0版本的代码.

本文参考文献如下:

- 讲解request的文章: http://blog.csdn.net/sbt0198/article/details/66479510
- cheerio中文api： https://cnodejs.org/topic/5203a71844e76d216a727d2e
- 讲解fs的文章： http://www.jianshu.com/p/5683c8a93511