[python爬虫scrapy的LinkExtractor](http://www.cnblogs.com/lei0213/p/8097515.html)

使用背景：

　　我们通常在爬去某个网站的时候都是爬去每个标签下的某些内容，往往一个网站的主页后面会包含很多物品或者信息的详细的内容，我们只提取某个大标签下的某些内容的话，会显的效率较低，大部分网站的都是按照固定套路（也就是固定模板，把各种信息展示给用户），LinkExtrator就非常适合整站抓取，为什么呢？因为你通过xpath、css等一些列参数设置，拿到整个网站的你想要的链接，而不是固定的某个标签下的一些链接内容，非常适合整站爬取。

[复制代码](javascript:void(0);)

1 import scrapy

2 from scrapy.linkextractor import LinkExtractor

3

4 class WeidsSpider(scrapy.Spider):

5 name = "weids"

6 allowed\_domains = ["wds.modian.com"]

7 start\_urls = ['http://www.gaosiedu.com/gsschool/']

8

9 def parse(self, response):

10 link = LinkExtractor(restrict\_xpaths='//ul[@class="cont\_xiaoqu"]/li')

11 links = link.extract\_links(response)

12 print(links)

[复制代码](javascript:void(0);)

links是一个list

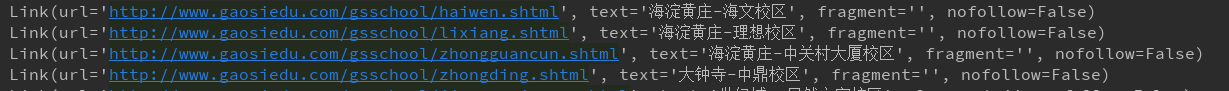
https://images2017.cnblogs.com/blog/976930/201712/976930-20171224103407850-1373089969.png

我们来迭代一下这个list

1 for link in links:

2 print(link)

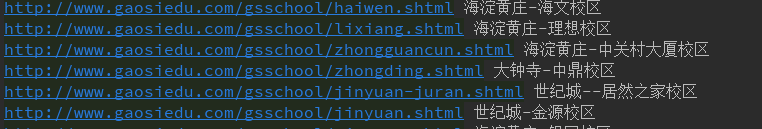
 links里面包含了我们要提取的url，那我们怎么才能拿到这个url呢？



直接在for循环里面link.url就能拿到我们要的url和text信息

1 for link in links:

2 print(link.url,link.text)



别着急，LinkExtrator里面不止一个xpath提取方法，还有很多参数。

>allow：接收一个正则表达式或一个正则表达式列表，提取绝对url于正则表达式匹配的链接，如果该参数为空，默认全部提取。

[复制代码](javascript:void(0);)

1 # -\*- coding: utf-8 -\*-

2 import scrapy

3 from scrapy.linkextractor import LinkExtractor

4

5 class WeidsSpider(scrapy.Spider):

6 name = "weids"

7 allowed\_domains = ["wds.modian.com"]

8 start\_urls = ['http://www.gaosiedu.com/gsschool/']

9

10 def parse(self, response):

11 pattern = '/gsschool/.+\.shtml'

12 link = LinkExtractor(allow=pattern)

13 links = link.extract\_links(response)

14 print(type(links))

15 for link in links:

16 print(link)

[复制代码](javascript:void(0);)

>deny：接收一个正则表达式或一个正则表达式列表，与allow相反，排除绝对url于正则表达式匹配的链接，换句话说，就是凡是跟正则表达式能匹配上的全部不提取。

[复制代码](javascript:void(0);)

1 # -\*- coding: utf-8 -\*-

2 import scrapy

3 from scrapy.linkextractor import LinkExtractor

4

5 class WeidsSpider(scrapy.Spider):

6 name = "weids"

7 allowed\_domains = ["wds.modian.com"]

8 start\_urls = ['http://www.gaosiedu.com/gsschool/']

9

10 def parse(self, response):

11 pattern = '/gsschool/.+\.shtml'

12 link = LinkExtractor(deny=pattern)

13 links = link.extract\_links(response)

14 print(type(links))

15 for link in links:

16 print(link)

[复制代码](javascript:void(0);)

>allow\_domains：接收一个域名或一个域名列表，提取到指定域的链接。

[复制代码](javascript:void(0);)

1 # -\*- coding: utf-8 -\*-

2 import scrapy

3 from scrapy.linkextractor import LinkExtractor

4

5 class WeidsSpider(scrapy.Spider):

6 name = "weids"

7 allowed\_domains = ["wds.modian.com"]

8 start\_urls = ['http://www.gaosiedu.com/gsschool/']

9

10 def parse(self, response):

11 domain = ['gaosivip.com','gaosiedu.com']

12 link = LinkExtractor(allow\_domains=domain)

13 links = link.extract\_links(response)

14 print(type(links))

15 for link in links:

16 print(link)

[复制代码](javascript:void(0);)

>deny\_domains：和allow\_doains相反，拒绝一个域名或一个域名列表，提取除被deny掉的所有匹配url。

[复制代码](javascript:void(0);)

1 # -\*- coding: utf-8 -\*-

2 import scrapy

3 from scrapy.linkextractor import LinkExtractor

4

5 class WeidsSpider(scrapy.Spider):

6 name = "weids"

7 allowed\_domains = ["wds.modian.com"]

8 start\_urls = ['http://www.gaosiedu.com/gsschool/']

9

10 def parse(self, response):

11 domain = ['gaosivip.com','gaosiedu.com']

12 link = LinkExtractor(deny\_domains=domain)

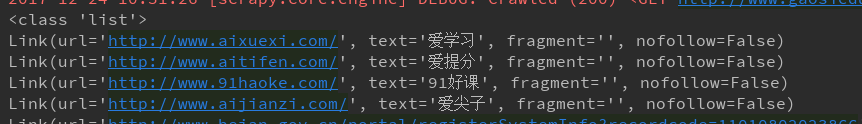
13 links = link.extract\_links(response)

14 print(type(links))

15 for link in links:

16 print(link)

[复制代码](javascript:void(0);)



>restrict\_xpaths：我们在最开始做那个那个例子，接收一个xpath表达式或一个xpath表达式列表，提取xpath表达式选中区域下的链接。

>restrict\_css：这参数和restrict\_xpaths参数经常能用到，所以同学必须掌握，个人更喜欢xpath。

[复制代码](javascript:void(0);)

1 # -\*- coding: utf-8 -\*-

2 import scrapy

3 from scrapy.linkextractor import LinkExtractor

4

5 class WeidsSpider(scrapy.Spider):

6 name = "weids"

7 allowed\_domains = ["wds.modian.com"]

8 start\_urls = ['http://www.gaosiedu.com/gsschool/']

9

10 def parse(self, response):

11 link = LinkExtractor(restrict\_css='ul.cont\_xiaoqu > li')

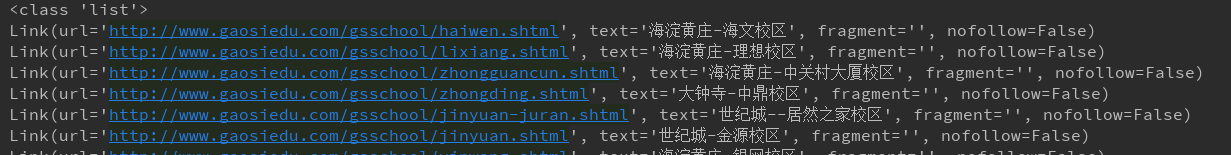
12 links = link.extract\_links(response)

13 print(type(links))

14 for link in links:

15 print(link)

[复制代码](javascript:void(0);)



>tags：接收一个标签（字符串）或一个标签列表，提取指定标签内的链接，默认为tags=（‘a’，‘area’）

>attrs：接收一个属性（字符串）或者一个属性列表，提取指定的属性内的链接，默认为attrs=（‘href’，），示例，按照这个中提取方法的话，这个页面上的某些标签的属性都会被提取出来，如下例所示，这个页面的a标签的href属性值都被提取到了。

[复制代码](javascript:void(0);)

1 # -\*- coding: utf-8 -\*-

2 import scrapy

3 from scrapy.linkextractor import LinkExtractor

4

5 class WeidsSpider(scrapy.Spider):

6 name = "weids"

7 allowed\_domains = ["wds.modian.com"]

8 start\_urls = ['http://www.gaosiedu.com/gsschool/']

9

10 def parse(self, response):

11 link = LinkExtractor(tags='a',attrs='href')

12 links = link.extract\_links(response)

13 print(type(links))

14 for link in links:

15 print(link)

[复制代码](javascript:void(0);)

前面我们讲了这么多LinkExtractor的基本用法，上面的只是为了快速试验，真正的基本用法是结合Crawler和Rule,代码如下

[复制代码](javascript:void(0);)

1 # -\*- coding: utf-8 -\*-

2 import scrapy

3 from scrapy.linkextractor import LinkExtractor

4 from scrapy.spiders.crawl import CrawlSpider,Rule

5

6

7 class GaosieduSpider(CrawlSpider):

8 name = "gaosiedu"

9 allowed\_domains = ["www.gaosiedu.com"]

10 start\_urls = ['http://www.gaosiedu.com/']

11 restrict\_xpath = '//ul[@class="schoolList clearfix"]'

12 allow = '/gsschool/.+\.shtml'

13 rules = {

14 Rule(LinkExtractor(restrict\_xpaths=restrict\_xpath), callback="parse\_item", follow=True)

15 }

16

17 def parse\_item(self,response):

18 schooll\_name = response.xpath('//div[@class="area\_nav"]//h3/text()').extract\_first()

19 print(schooll\_name)

[复制代码](javascript:void(0);)

简单的说一下，上面我们本应该继承scrapy.Spider类，这里需要继承CrawlSpider类（因为CrawlSpider类也是继承了scrapy.Spider类），rules是基本写法，可不是随便哪个单词都可以的啊，而且注意rules必须是一个list或者dict，如果是tuple的话就会报错。里面的话Rule里面包含了几个参数，LinkExtractor就不在这里熬述了，看上面就行，至于其他的几个参数，可以看我们另外一篇博文：http://www.cnblogs.com/lei0213/p/7976280.html