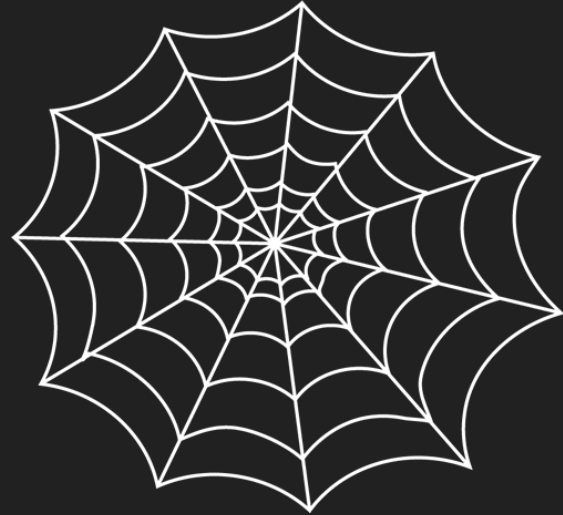
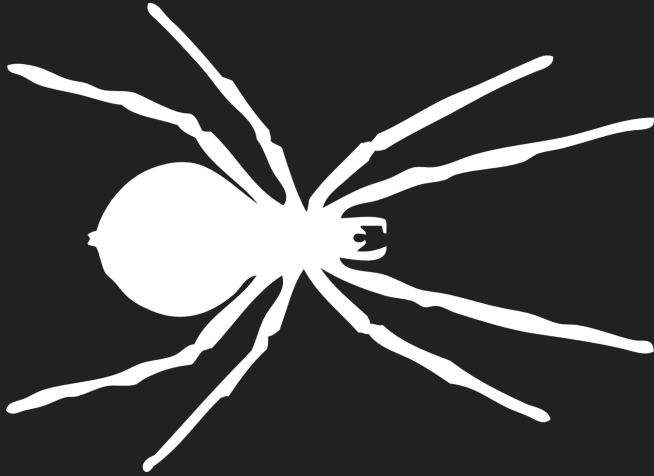


Web scraping

with Scrapy

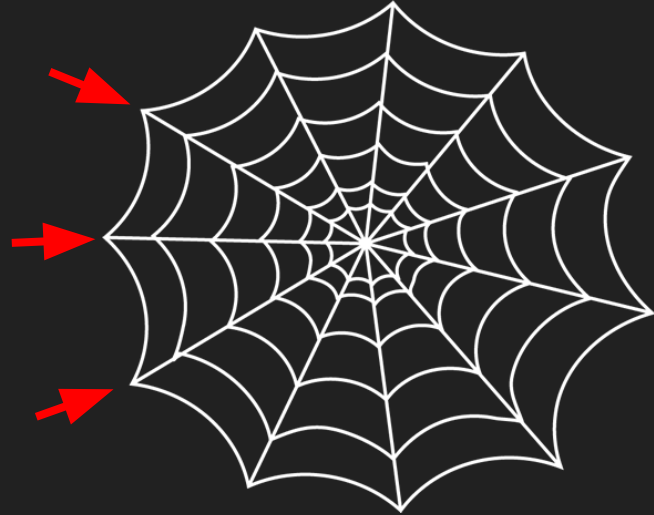
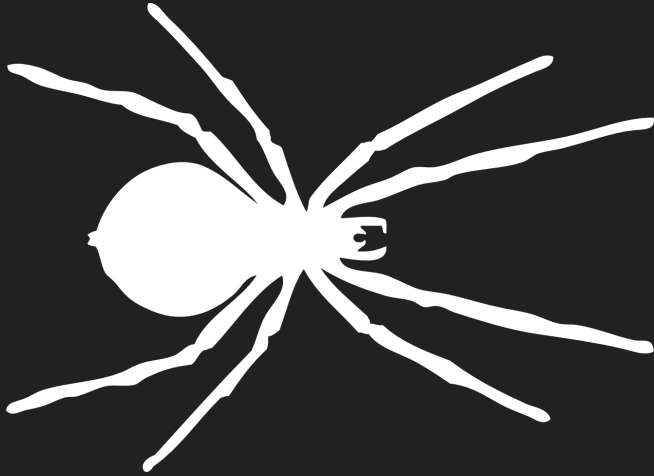
Web crawler

a program that systematically browses the web



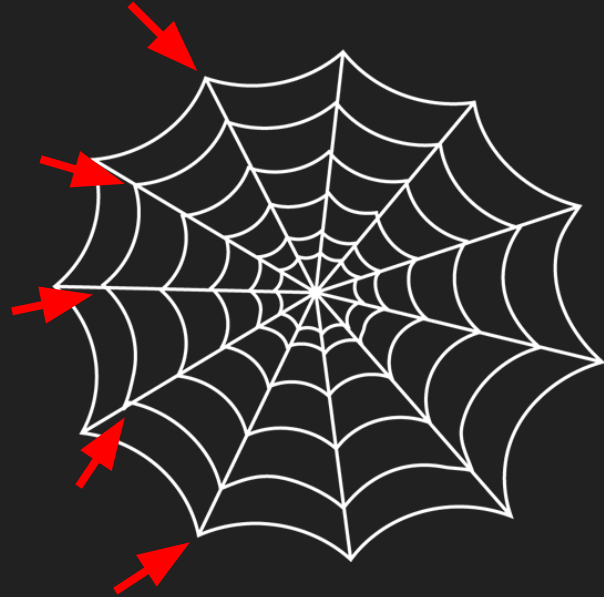
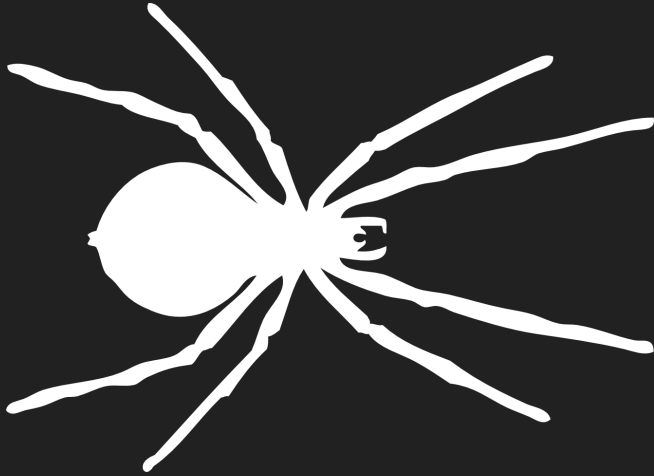
Web crawler

starts with list of URLs to visit (seeds)



Web crawler

identifies links, adds them to list of URLs to visit



Uses

Scraping



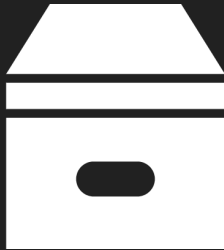
Indexing



Spambots



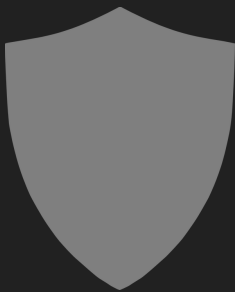
Archiving



Validation



Crawling policy and robots.txt



Selection policy

Revisit policy

Politeness policy

Parallelization policy

Examples

Google



DuckDuckGo

YAHOO!



Scrapy

web scraping framework for Python


```
pip install scrapy
```

Creating a project

```
scrapy startproject tutorial
```

```
tutorial
```

```
|— scrapy.cfg
|— tutorial
    |— __init__.py
    |— items.py
    |— pipelines.py
    |— settings.py
    |— spiders
        |— __init__.py
```

Creating a spider

```
cd tutorial/tutorial/spiders  
touch bwog_spider.py  
open bwog_spider.py
```

Creating a spider

```
import scrapy

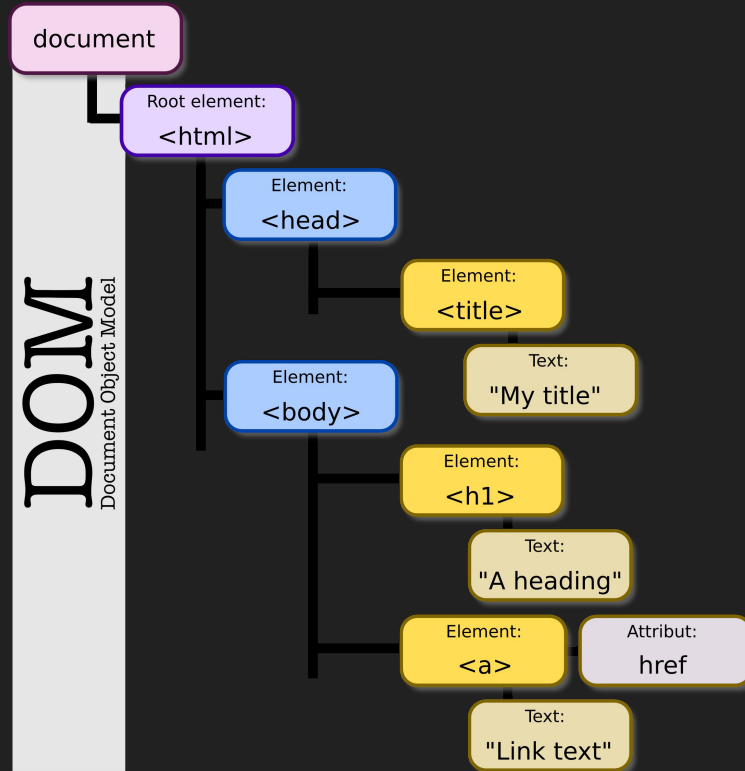
class BwogSpider(scrapy.Spider):
    name = 'bwog'
    allowed_domains = ['bwog.com']
    start_urls = ['http://bwog.com/']

    def parse(self, response):
        print 'Testing ' + response.url
```

Running the spider

```
cd ../../  
scrapy crawl bwog
```

Structure of a document



XPath

Language for **selecting nodes** in XML documents

```
/wikimedia/projects/project/editions/*[2]
```

```
<?xml version="1.0" encoding="utf-8"?>
<wikimedia>
  <projects>
    <project name="Wikipedia" launch="2001-01-05">
      <editions>
        <edition language="English">en.wikipedia.org</edition>
        <edition language="German">de.wikipedia.org</edition>
        <edition language="French">fr.wikipedia.org</edition>
        <edition language="Polish">pl.wikipedia.org</edition>
      </editions>
    </project>
    <project name="Wiktionary" launch="2002-12-12">
      <editions>
        <edition language="English">en.wiktionary.org</edition>
        <edition language="French">fr.wiktionary.org</edition>
        <edition language="Vietnamese">vi.wiktionary.org</edition>
        <edition language="Trukish">tr.wiktionary.org</edition>
      </editions>
    </project>
  </projects>
</wikimedia>
```

CSS Selectors

Selects nodes based on their CSS style

Each stylesheet rule has a **selector pattern** that matches a set of HTML elements

```
p{  
  color:#0000FF;  
}
```

```
<h1 class="class1">  
  Heading  
</h1>  
<p id="id1">  
  Paragraph  
</p>  
<p class="class1">  
  Paragraph  
</p>
```

```
.class1{  
  color:#0000FF;  
}
```

```
<h1 class="class1">  
  Heading  
</h1>  
<p id="id1">  
  Paragraph  
</p>  
<p class="class1">  
  Paragraph  
</p>
```

```
#id1{  
  color:#0000FF;  
}
```

```
<h1 class="class1">  
  Heading  
</h1>  
<p id="id1">  
  Paragraph  
</p>  
<p class="class1">  
  Paragraph  
</p>
```

```
p.class1{  
  color:#0000FF;  
}
```

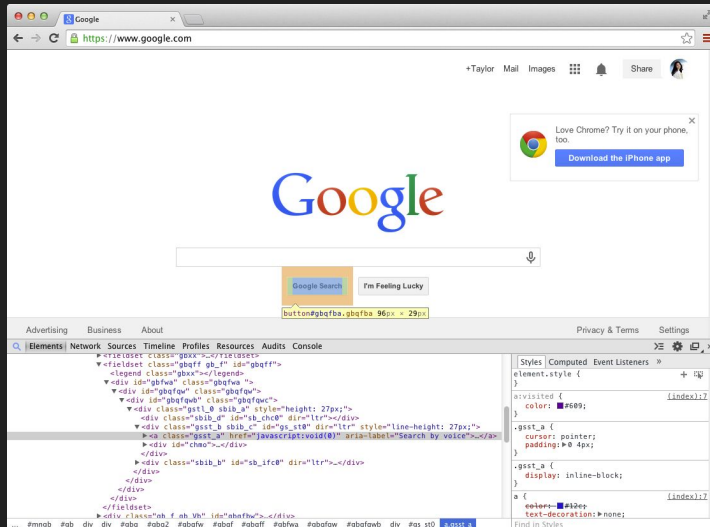
```
<h1 class="class1">  
  Heading  
</h1>  
<p id="id1">  
  Paragraph  
</p>  
<p class="class1">  
  Paragraph  
</p>
```


Obtaining the XPath



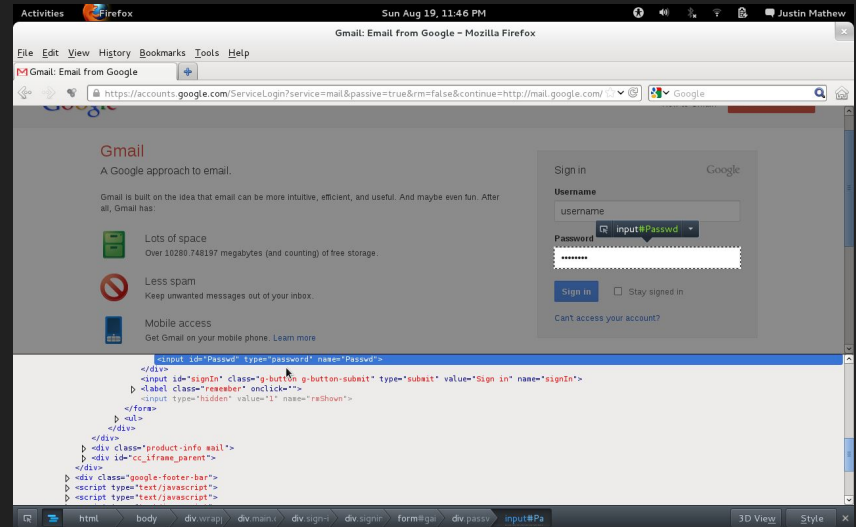
Chrome Developer Tools

Inspect
Copy > Copy XPath



Firebug HTML Panel

Inspect Element with Firebug
Copy XPath



Using XPath selectors

The screenshot shows a web browser with a right-click context menu open over a link titled "Junior Phi Beta Kappa Inductees Announced". The menu options include: "Open Link in New Tab", "Open Link in New Window", "Open Link in Incognito Window", "Save Link As...", "Copy Link Address", "Copy", "Search Google for 'Junior Phi Beta Kappa Inductees Announced'", "Print...", "Inspect", "Speech", "Search With Google", and "Add to iTunes as a Spoken Track". The background shows a snippet of an article with the text "The 23 students" and a list of names.

Jan 6 15

Junior Phi Beta Kappa Inductees Announced

Written by B

January 06

Yesterday afternoon, Columbia College released the names of the 23 members of the Class of 2016 senior class who were chosen to be inducted into the national honor society this year, these 23 students make up just 2%; the other 8% will be inducted in the Spring term. The students were chosen by a faculty committee of PBK members "based on the breadth, depth and rigor of their academic programs, as well as recommendations from faculty members who have worked with closely them." Congratulations early inductees—you're the top 2%!

The 23 students

- Karen Bao
- Katherine Christianson
- Emily Corning
- Bernhard Fasenfest
- Adriano Fernandes
- Vahe Galstyan
- Elena Goldstein

Your prize

The screenshot shows a web browser displaying the article "Junior Phi Beta Kappa Inductees Announced" by BWOG STAFF. The article text is visible, along with a photo of a Phi Beta Kappa key. The browser's developer tools are open, showing the "Elements" panel with the following HTML structure:

```
<div class="center-section">
  <div class="col-sm-1 col-md-1 col-xs-2 padding0"></div>
  <div class="col-sm-11 col-md-11 col-xs-10">
    <div class="blog-section">
      <a href="http://bwog.com/2016/01/06/junior-phi-beta-kappa-inductees-announced/" title="Junior Phi Beta Kappa Inductees Announced">Junior Phi Beta Kappa Inductees Announced</a>
    </div>
    <div class="written-by"></div>
    <div class="comment-time"></div>
    <div class="blog-text"></div>
  </div>
</div>
```

The "Styles" panel on the right shows the default styles for the selected element, including float, width, and border-bottom.

Paying the Bills

Take NY Defensive Driving all online

Jan 6 15

Junior Phi Beta Kappa Inductees Announced

Written by BWOG STAFF

January 06, 2016 12:11 pm 15 Comments

Yesterday afternoon, Columbia College released the names of the 23 members of the Class of 2016 being inducted into Phi Beta Kappa for the Fall term. Of the 10% of the senior class who will be inducted into the national honor society this year, these 23 students make up just 2%; the other 8% will be inducted in the Spring term. The students were chosen by a faculty committee of PBK members "based on the breadth, depth and rigor of their academic programs, as well as recommendations from faculty members who have worked with closely them." Congratulations early inductees—you're the top 2%!

2 999

Elements Console Sources Network Timeline Profiles Resources Audits

Styles Computed Event Listeners

Filter

element.style { }

.blog-section { float: left; width: 100%; border-bottom: 1px solid #5d5d5d; }

media="all" body, div, dl, dt, foundation.css?ver=4.3.2:44 ds, ul, ol, li, h1, h2, h3, h4, h5, h6, pre, form, p, blockquote, th, td { }

`//div[@class="blog-section"]`

Using XPath selectors

```
import scrapy

class BwogSpider(scrapy.Spider):
    name = 'bwog'
    allowed_domains = ['bwog.com']
    start_urls = ['http://bwog.com/']

    def parse(self, response):
        for section in response.xpath('//div[@class="blog-section"]'):
            title = section.xpath('.//a/text()').extract()[0]
            print title
```

Running the spider

scrapy crawl bwog

Junior Phi Beta Kappa Inductees Announced

Until Next Year, Columbia

Overseen: Historical Revisionism On The Columbia Facebook Page

Bwog's Resolutions For The New Year

Bwog In Bed: New Level Edition

Actual Wisdom: Philip Protter

Bwog Poetry

Bunsen Bwog: If This Is The Apocalypse, At Least We'll Find Love

Bwog In Bed: Is This The Real Life? Edition

Field Notes: Barely Hanging On Edition

Actual Wisdom: J.C. Salyer

Shit Our Notes Say

"Baking" With Bwog: The Many Uses of Oregano

Bwog In Bed: Be Your Own Hero Edition

Dark Night Of The (Final) Soul

Scrapy Requests

```
import scrapy

class BwogSpider(scrapy.Spider):
    name = 'bwog'
    allowed_domains = ['bwog.com']
    start_urls = ['http://bwog.com/']

    def parse(self, response):
        for section in response.xpath('//div[@class="blog-section"]'):
            link = section.xpath('.//a/@href').extract()[0]
            yield scrapy.Request(link, callback=self.parse_entry)

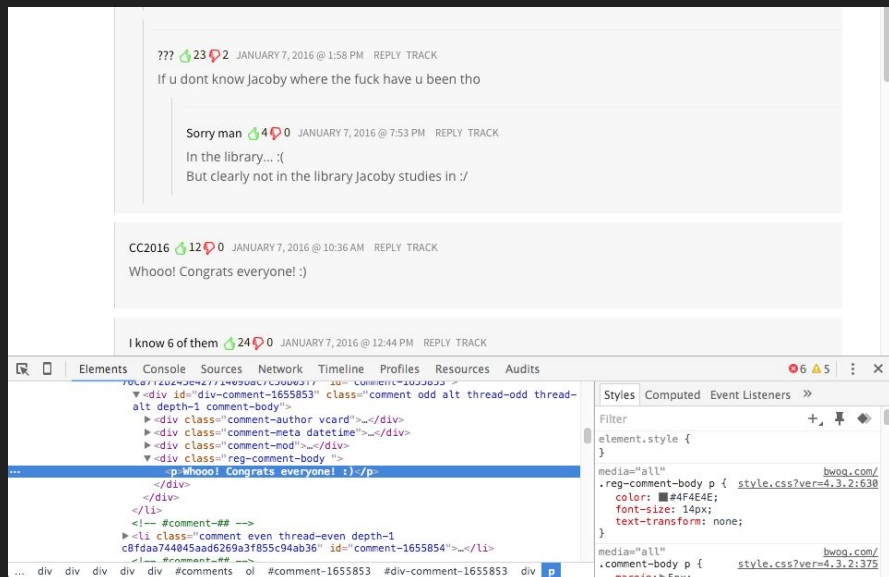
    def parse_entry(self, response):
        print response.url
```

Running the spider

```
scrapy crawl bwog
```

```
http://bwog.com/2016/01/06/junior-phi-beta-kappa-inductees-announced/  
http://bwog.com/2015/12/23/until-next-year-columbia/  
http://bwog.com/2015/12/23/overseen-historical-revisionism-on-the-columbia-facebook-page/  
http://bwog.com/2015/12/23/bwogs-resolutions-for-the-new-year/  
http://bwog.com/2015/12/23/bwog-in-bed-new-level-edition/  
http://bwog.com/2015/12/22/actual-wisdom-philip-protter/  
http://bwog.com/2015/12/22/bwog-poetry/  
http://bwog.com/2015/12/22/bunsen-bwog-if-this-is-the-apocalypse-at-least-we'll-find-love/  
http://bwog.com/2015/12/22/bwog-in-bed-is-this-the-real-life-edition/  
http://bwog.com/2015/12/21/field-notes-barely-hanging-on-edition/  
http://bwog.com/2015/12/21/actual-wisdom-j-c-salyer/  
http://bwog.com/2015/12/21/shit-our-notes-say/  
http://bwog.com/2015/12/21/baking-with-bwog-the-many-uses-of-oregano/  
http://bwog.com/2015/12/21/bwog-in-bed-be-your-own-hero-edition/  
http://bwog.com/2015/12/20/dark-night-of-the-final-soul/
```

Extracting comments



```
//div[@class="reg-comment-body "]
```

Extracting comments

```
import scrapy

class BwogSpider(scrapy.Spider):
    name = 'bwog'
    allowed_domains = ['bwog.com']
    start_urls = ['http://bwog.com/']

    def parse(self, response):
        for section in response.xpath('//div[@class="blog-section"]'):
            link = section.xpath('.//a/@href').extract()[0]
            yield scrapy.Request(link, callback=self.parse_entry)

    def parse_entry(self, response):
        for comment in response.xpath('//div[@class="reg-comment-body " ]'):
            paragraphs = comment.xpath('.//text()').extract()
            text = '\n'.join(paragraphs).strip()
            print text
```


Scrapy Items

```
open tutorial/items.py
```

Scrapy Items

```
import scrapy

class TutorialItem(scrapy.Item):
    # define the fields for your item here like:
    # name = scrapy.Field()
    content = scrapy.Field()
```

Extracting comments

```
import scrapy
from tutorial.items import TutorialItem

class BwogSpider(scrapy.Spider):
    name = 'bwog'
    allowed_domains = ['bwog.com']
    start_urls = ['http://bwog.com/']

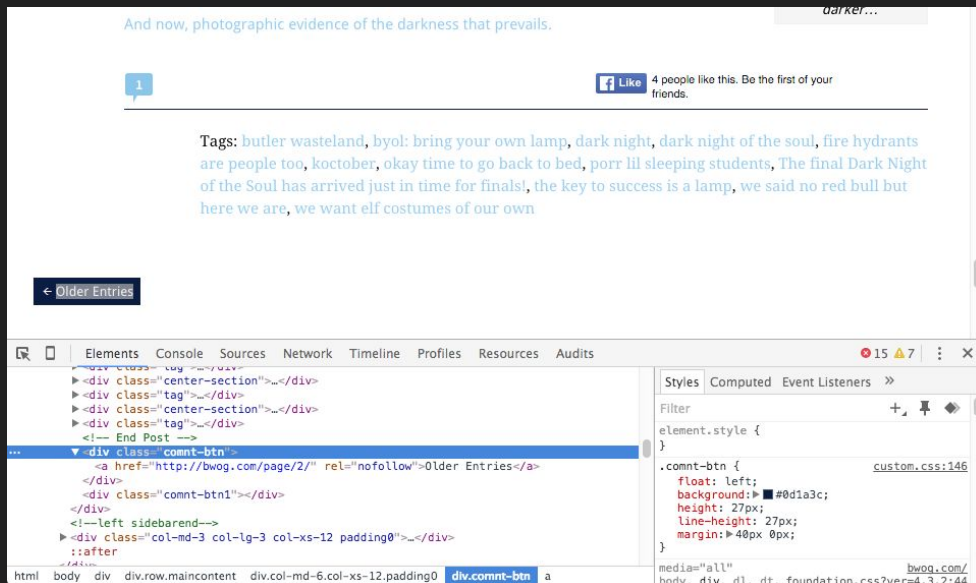
    def parse(self, response):
        for section in response.xpath('//div[@class="blog-section"]'):
            link = section.xpath('.//a/@href').extract()[0]
            yield scrapy.Request(link, callback=self.parse_entry)

    def parse_entry(self, response):
        for comment in response.xpath('//div[@class="reg-comment-body " ]'):
            paragraphs = comment.xpath('.//text()').extract()
            text = '\n'.join(paragraphs).strip()
            item = TutorialItem()
            item['content'] = text
            yield item
```

Storing comments

```
scrapy crawl bwog -o comments.json
```

Following links



```
//div[@class="comnt-btn"]//@href
```

Following links

```
import scrapy
from tutorial.items import TutorialItem

class BwogSpider(scrapy.Spider):
    name = 'bwog'
    allowed_domains = ['bwog.com']
    start_urls = ['http://bwog.com/']

    def parse(self, response):
        for section in response.xpath('//div[@class="blog-section"]'):
            link = section.xpath('.//a/@href').extract()[0]
            yield scrapy.Request(link, callback=self.parse_entry)
            next = response.xpath('//div[@class="comnt-btn"]//a/@href').extract()[0]
            yield scrapy.Request(next, callback=self.parse)

    def parse_entry(self, response):
        for comment in response.xpath('//div[@class="reg-comment-body "]'):
            paragraphs = comment.xpath('.//text()').extract()
            text = '\n'.join(paragraphs).strip()
            item = TutorialItem()
            item['content'] = text
            yield item
```

Wait for it...

```
scrapy crawl bwog -o comments.json
```

...and on it goes!

Next steps

Scrapy Shell (interactive shell console)

Feed Exports (JSON, CSV, XML)

Item Pipelines (writing to MongoDB, duplicates filter)

doc.scrapy.org

for more information