https://tanpham.org

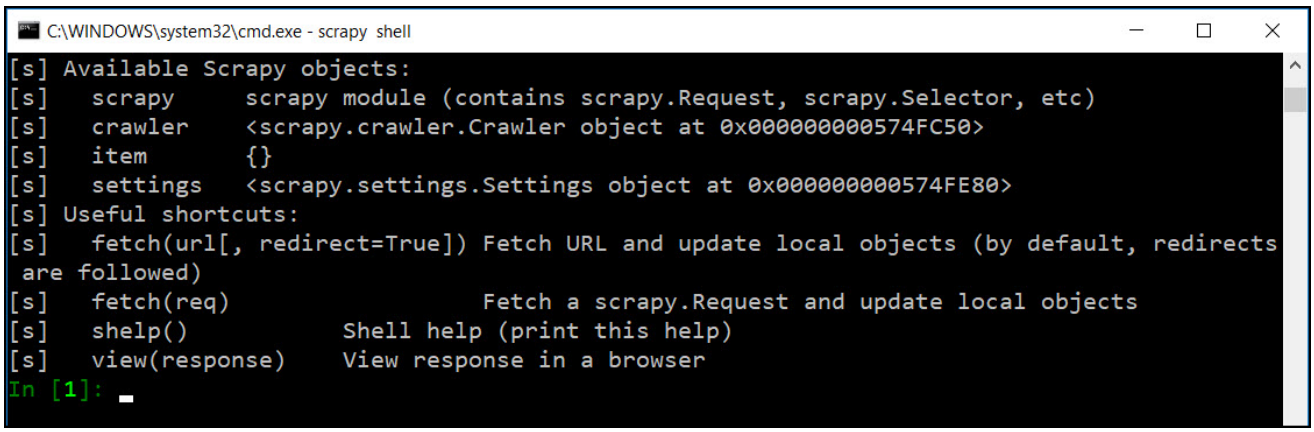
Explain **Shell**, **Spider**, **Item**, **Crawl**, **ItemPipeline** and **Recursive Spider**.

# Milestone 1 : scrape titles, links, score in one page


One of my favorite on Reddit is funny image page, you could access at https://www.reddit.com/r/funny/ . First part of this tutorial will explain how to scrape the link, title and score from above link.

# Shell : understanding the page

Scrapy framework include a very handy tool called `shell` . With `shell` you could try to fetch url, then try to extract data from `response` object. To access `shell` , from command prompt typing in `scrapy shell` , now the shell ready to accept your commands.




Typing following command then enter

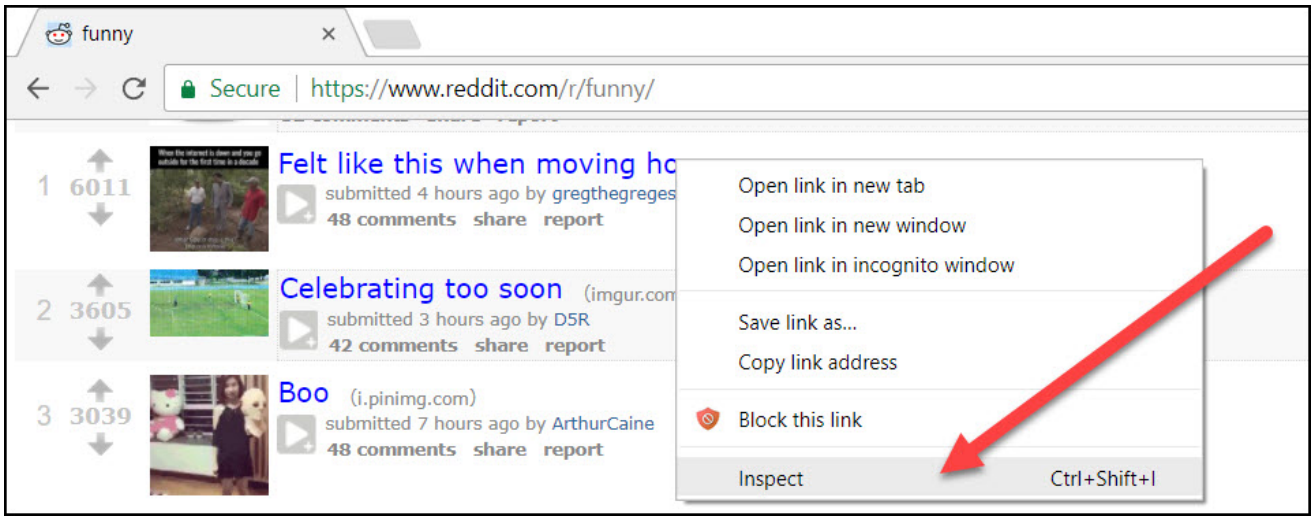
```
fetch('https://www.reddit.com/r/funny/')
```

After execute above command, a object call `response` is created, `response` object represent result of command `fetch` , so it contain all data from above Reddit url. After have `response` object, we could show up some information as below:

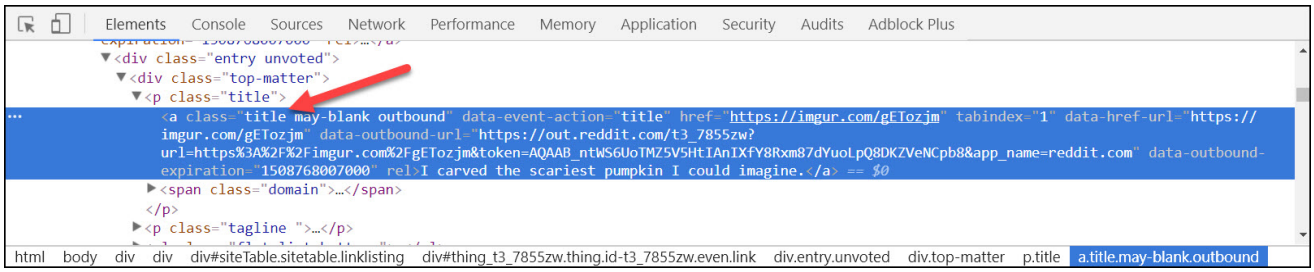
- Typing in `view(response)` , a local HTML page is show up on browser, allow us visually know what contain in the page.
- Typing in `response.url` , this command will show up original url.
- Typing in `response.text` , this command show up whole HTML source code for this page.

The main thing we need to tell to Scrapy is how to extract data from `response` object. Have 2 way to extract data, using `css selector` or `xpath` . In this tutorial we will use `css selector` .

From Chrome browser, open url https://www.reddit.com/r/funny/ , move your mouse above one of title , right click and select `inspect`




From inspection tool, we see that we need to care about all `a` tag which has css class call `title`




Type following command then enter :

```
response.css("a.title").extract()
```

This command using css selector to extract all `a` tags which has class `title` .


We want text title, so typing following command then enter

```
response.css("a.title::text").extract()
```

This command will extract all wanted title text as show below

```
In [17]: response.css("a.title::text").extract()
Out[17]:
[u'Subreddit Of The Month [October 2017]: /r/oneliners. Know of a small (under 20,000 subscribers) humor-based subreddit that deserves a month in the spotlight? Link it inside!',
 u'I carved the scariest pumpkin I could imagine.',
 u"it's not just Ahmed..",
 u'Told my 3 and 4 yr olds to put the toilet paper under the sink... must be more specific next time',
 u'Yesterday my little sister said she was going to make a walking stick, I assumed she meant one for helping you walk, but today I found this on the front porch',
 u'Work perks',
 u'My three dads?',
 u'An internet data security manager is seen hard at work',
 u'I got surprise roasted on the road today.',
 u'Blazing Saddles',
```

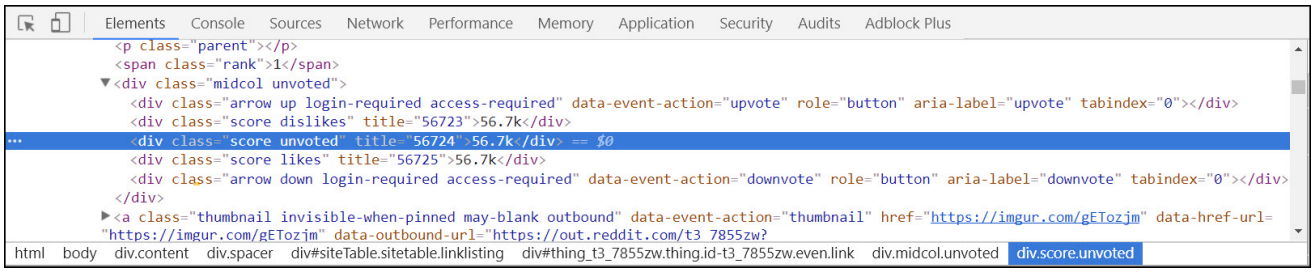
To extract the link title, from inspection tool, we need to get value of attribute name `href`

```
response.css("a.title::attr(href)").extract()
```

All links will be extracted

```
In [18]: response.css("a.title::attr(href)").extract()
Out[18]:
[u'https://www.reddit.com/r/oneliners/',
 u'https://imgur.com/gETozjm',
 u'/r/funny/comments/7868dq/its_not_just_ahmed/',
 u'/r/funny/comments/784brb/told_my_3_and_4_yr_olds_to_put_the_toilet_paper/',
 u'/r/funny/comments/785b6j/yesterday_my_little_sister_said_she_was_going_to/',
 u'/r/funny/comments/7836pp/work_perks/',
 u'/r/funny/comments/7830z5/my_three_dads/',
 u'http://i.imgur.com/EUrN98R.gifv',
 u'/r/funny/comments/783oro/i_got_surprise_roasted_on_the_road_today/',
 u'https://i.imgur.com/h5qRoVu.jpg',
 u'https://i.imgur.com/NOjuZ9z.gifv',
 u'https://i.imgur.com/MQksv34.gifv',
 u'https://i.imgur.com/t7H8Ato.gifv',
 u'https://i.imgur.com/HK7pDjo.gifv',
 u'/r/funny/comments/78340d/while_we_all_focused_on_the_iphone_x/',
 u'/r/funny/comments/784zrk/its_rover_raggy/',
 u'https://i.imgur.com/9DtEbSB.gif',
 u'/r/funny/comments/780xhp/there_are_3_types_of_girls_on_halloween/',
 u'https://i.imgur.com/tleQFKU.gifv',
 u'/r/funny/comments/785vqh/mans_best_friend/',
 u'/r/funny/comments/780sjd/this_guys_shirt_this_morning_in_traffic/',
 u'/r/funny/comments/781c9y/my_dog_has_3_legs_i_tried_to_make_the_most_of_it/',
 u'/r/funny/comments/786pjs/ranch_day_yum_xpost_rteenagers/',
 u'/r/funny/comments/780scn/got_chewed_out_by_my_mom_because_im_a_pot_head/',
 u'https://gfycat.com/handsomehandyaardvark',
 u'https://imgur.com/SCAL8MQ.gifv']
```

Related to score, let's do another inspection by move mouse to above score, right click and then select `inspect` . We found that we need to find `div` tag with class `score unvoted` .




Let's try to extract all score by following command

```
response.css("div.score.unvoted::attr(title)").extract()
```

Following result return

```
In [19]: response.css("div.score.unvoted::attr(title)").extract()
Out[19]:
[u'446',
 u'56720',
 u'2738',
 u'12133',
 u'2942',
 u'80763',
 u'22677',
 u'5409',
 u'5681',
 u'523',
 u'7536',
```

This understanding of how to extract data will be completely applied when we create Scrapy project. Now let summary useful thing we can do with shell:

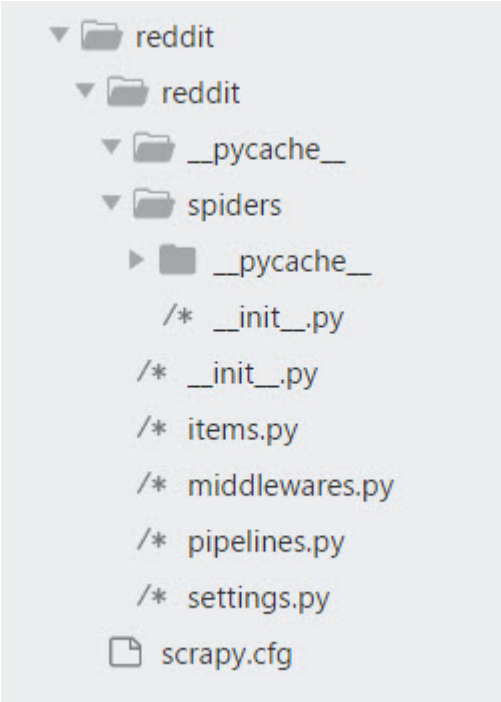
- ***fetch('url')*** will return `response` object which contain all information.
- ***view(response)*** view local web page on browser
- ***response.url*** will return original url which using on fetch command
- ***response.text*** will return entire HTML source code from page
- ***response.css('').extract()*** will filter HTML source code based on css selector then extract wanted information
  - *'css_selector::**text**'* will return text of extracted tags
  - *'css_selector::**attr**(attribute_name)'* will return value of attribute_name

# Spider : where to start and how to extract

Now, it time to create new Scrapy project. From command prompt, enter command and enter

```
scrapy startproject reddit
```

A new project folder name **reddit** will be automatically created with following structure




Now change current directory to **reddit** then create a new `spider` . You need to pass on spider name and the domain, in this case is reddit.com

```
cd reddit
scrapy genspider reddit_job reddit.com
```

A new file call **reddit_job.py** will be created inside spider folder with following content

```
# -*- coding: utf-8 -*-
import scrapy

class RedditJobSpider(scrapy.Spider):
    name = 'reddit_job'
    allowed_domains = ['reddit.com']
    start_urls = ['http://reddit.com/']

    def parse(self, response):
        pass
```

Let explain what inside this spider file

- `name` is the name of spider, this name will be used when we want to run this spider
- `allowed_domains` spider will only crawl file in this list of domain
- `start_urls` starting point so spider start crawling, let edit the `start_urls` to https://www.reddit.com/r/funny/
- `parse` function, this function will parse the response which return automatically from crawl result. We will use selector which already tried successfully with `shell` in this function to extract data.

Let try to change `parse` function as following code

```
# -*- coding: utf-8 -*-
import scrapy


class RedditJobSpider(scrapy.Spider):

    name = 'reddit_job'
    allowed_domains = ['reddit.com']

    # we start from funny title
    start_urls = ['https://www.reddit.com/r/funny/']

    def parse(self, response):
        print response.css("a.title::text").extract()
        print response.css("a.title::attr(href)").extract()
        print response.css("div.score.unvoted::attr(title)").extract()
```

Now let back to command prompt and start the reddit_job spider with command

```
scrapy crawl reddit_job
```

You will see our data is extract and print out to console as 3 list. So, spider extraction working fine.

# Item : define what to extract

`items.py` will be place where you specify what data you want. Each data item will be a `Field` object.

Let change file `items.py` with following content

```
# -*- coding: utf-8 -*-

# Define here the models for your scraped items
#
# See documentation in:
# http://doc.scrapy.org/en/latest/topics/items.html

import scrapy

class RedditItem(scrapy.Item):
    title = scrapy.Field()
    url = scrapy.Field()
    score = scrapy.Field()
```

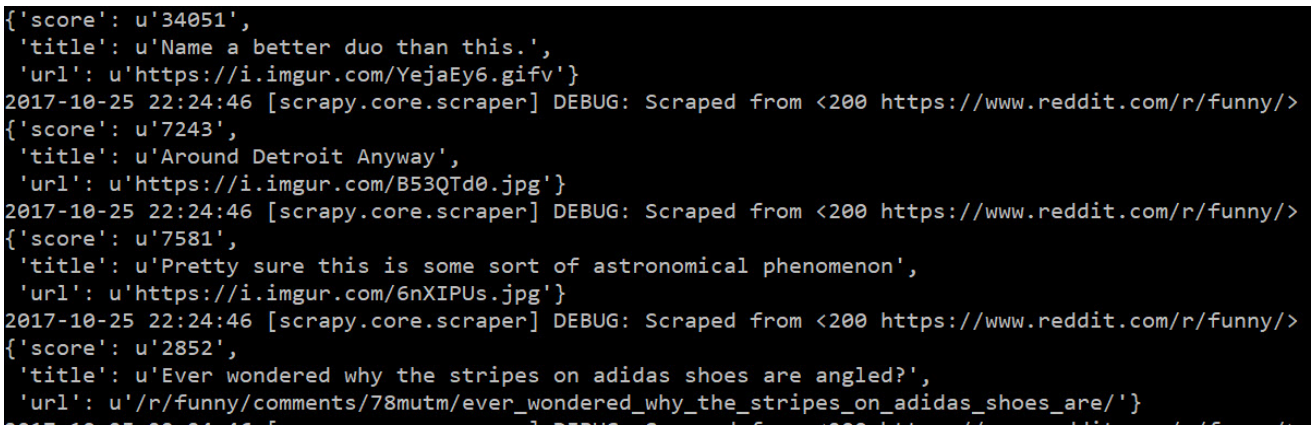
Now let change spider `parse` function a little bit correspond to item we just created. The point is `parse` function will `yield` out items, each item will contain data we want to extract.

```
# -*- coding: utf-8 -*-
import scrapy
# Import so parse could understand RedditItem define structure
from reddit.items import RedditItem

class RedditJobSpider(scrapy.Spider):

    name = 'reddit_job'
    allowed_domains = ['reddit.com']

    # we start from funny title
    start_urls = ['https://www.reddit.com/r/funny/']

    def parse(self, response):
        titles = response.css("a.title::text").extract()
        hrefs = response.css("a.title::attr(href)").extract()
        scores = response.css("div.score.unvoted::attr(title)").extract()

        for item in zip(titles, hrefs, scores):

            new_item = RedditItem()

            new_item['title'] = item[0]
            new_item['url'] = item[1]
            new_item['score'] = item[2]

            yield new_item
```

Now let's run the crawl command again

```
scrapy crawl reddit_job
```

Console will print out data items as expected. That it, we just create a fully functional Scrapy project.




# Crawl : store data to json, csv, xml file

For almost Scrapy project, after be extracted data will be saved to database or file. To save data to csv file, let execute crawl command as below

```
scrapy crawl reddit_job -o out_data.csv -t csv
```

File with name `out_data.csv` is created and contain our data like magical.

| url | score | title |
|---|---|---|
| https://www.reddit.com/r/oneliners/ | 485 | Subreddit Of The Month [October 2017]: /r/oneliners. Know of a |
| https://i.imgur.com/B53QTd0.jpg | 9615 | Around Detroit Anyway |
| https://i.imgur.com/YejaEy6.gifv | 36473 | Name a better duo than this. |
| https://i.imgur.com/6nXIPUs.jpg | 8341 | Pretty sure this is some sort of astronomical phenomenon |
| /r/funny/comments/78mutm/ever_wondered_w | 3491 | Ever wondered why the stripes on adidas shoes are angled? |
| /r/funny/comments/78n9af/meanwhile_in_a_pa | 2181 | Meanwhile in a parallel universe... |
| /r/funny/comments/78kvcp/breaking_news/ | 20058 | Breaking news! |
| /r/funny/comments/78l6cp/dentists_waiting_ro | 6444 | Dentist's waiting room. Imagine a gynecologist using the same co |
| http://i.imgur.com/p56XzHG.gifv | 14016 | "Oh hey, you're home early" |
| https://i.imgur.com/jFuTlVK.png | 1394 | A useful book for your twenties |
| /r/funny/comments/78kl51/scissor_me/ | 5762 | Scissor me! |
| /r/funny/comments/78jaqp/this_fish_is_giving_a | 20799 | This fish is giving attitude after I moved his rocks |
| https://i.imgur.com/QLfINCV.gifv | 3951 | Kayaking during Halloween. |
| /r/funny/comments/78kj9z/take_your_kids_to_v | 4215 | Take Your Kids to Work Day |
| https://i.imgur.com/4GrV9bx.gifv | 5351 | Clever girl |
| /r/funny/comments/78j88z/sad_boi/ | 9719 | Sad boi |
| /r/funny/comments/78ithf/if_this_isnt_a_sign_i | 12906 | If this isnâ€™t a sign, I donâ€™t know what is. |
| https://i.imgur.com/7ONnImO.jpg | 3411 | Nani? |

It is very similar if you want export data to json or xml file

```
scrapy crawl reddit_job -o out_data.json -t json
```

```
scrapy crawl reddit_job -o out_data.xml -t xml
```

# Item Pipeline : filter with score value

Some time after extracting data (after `yield` in `parse` function) you want to do some extra processing before data go to data base or file. Some kind of extra processing : remove duplicate, add to database. In this session I will show you how to filter data base on score value. Mean if the score is below a value, I will drop data, so the data not go in to csv file.

To do this Scrapy provide object call `ITEM_PIPELINES` . Have 2 steps to enable and apply `ITEM_PIPELINES`. First, from reddit project folder , open the file `settings.py` then uncomment `ITEM_PIPELINES` part.

```
ITEM_PIPELINES = {
   'reddit.pipelines.RedditPipeline': 300,
}
```

Second, from project folder, open the file `pipelines.py` , in this file you will modify function `process_item` to add your logic you want. In this function, I will drop the data item if score value below 10000.

```
# -*- coding: utf-8 -*-

# Define your item pipelines here
#
# Don't forget to add your pipeline to the ITEM_PIPELINES setting
# See: http://doc.scrapy.org/en/latest/topics/item-pipeline.html

from scrapy.exceptions import DropItem

class RedditPipeline(object):
    def process_item(self, item, spider):
        if int(item['score']) > 10000:
            return item
        else:
            raise DropItem("too low score")
```

Now let try crawl command

```
scrapy crawl reddit_job -o out_data_filter.csv -t csv
```

From the console you will see drop message for data which have score < = 10000

```
2017-10-25 23:19:23 [scrapy.core.scraper] WARNING: Dropped: too low score
{'score': u'1802',
 'title': u"Move over kids, I'll show you know it's done",
 'url': u'https://imgur.com/fUE8qkU'}
```

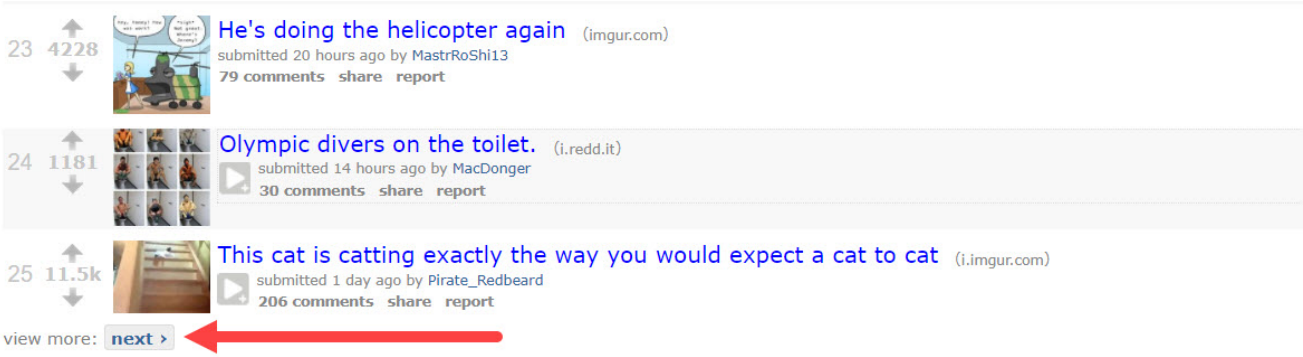
In csv file, you will only see the data item which score > 10000

| url | score | title |
|---|---|---|
| https://i.imgur.com/B53QTd0.jpg | 19524 | Around Detroit Anyway |
| https://i.imgur.com/YejaEy6.gifv | 38820 | Name a better duo than this. |
| https://i.imgur.com/6nXIPUs.jpg | 10011 | Pretty sure this is some sort of astronomical phenomenon |
| /r/funny/comments/78kvcp/breaking_new | 20730 | Breaking news! |
| http://i.imgur.com/p56XzHG.gifv | 14459 | "Oh hey, you're home early" |
| /r/funny/comments/78jaqp/this_fish_is_gi | 21023 | This fish is giving attitude after I moved his rocks |
| /r/funny/comments/78ithf/if_this_isnt_a_ | 13051 | If this isnâ€™t a sign, I donâ€™t know what is. |
| https://imgur.com/znW3AyR.gifv | 71463 | That's one way to have fun at work |
| https://i.imgur.com/rDFEavJ.jpg | 13580 | If you ever get stranded in your saucer |

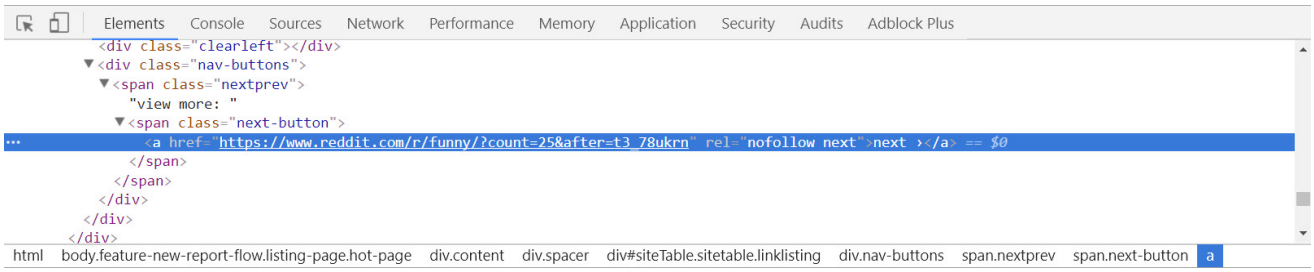
# Milestone 2 : scrape all pages with recursive spider

From beginning of this tutorial until now we only got data from one page https://www.reddit.com/r/funny/

but if you open this url by browser, you will see that still remain a lot more fun when you click to **Next** button at end of page. In this session I will show you how to scrape from all page.




Now let do some inspection about this **Next** button.




Open the `shell` and try with followings command

Fetch the url

```
fetch('https://www.reddit.com/r/funny/')
```

Select for `span` with class `next-button` .

```
response.css(".next-button")
```

Select for `a` tag inside next button and extract the link

```
response.css(".next-button").css("a::attr(href)").extract()[0]
```

Will return the link to the next page, that is what we want

```
In [6]: response.css(".next-button").css("a::attr(href)").extract()[0]
Out[6]: u'https://www.reddit.com/r/funny/?count=25&after=t3_78r6eg'
```

# Recursive spider : change parse function

From above session, we already know how to get the link to next page, we will need modify the `parse` function inside `spider` also. The main point is extract next page url and then make a `Request` with callback to `parse` function.

```
# -*- coding: utf-8 -*-
import scrapy
from reddit.items import RedditItem
from scrapy.http.request import Request

class RedditJobSpider(scrapy.Spider):

    name = 'reddit_job'
    allowed_domains = ['reddit.com']

    # we start from funny title
    start_urls = ['https://www.reddit.com/r/funny/']

    def parse(self, response):

        # extract data from raw response
        titles = response.css("a.title::text").extract()
        hrefs = response.css("a.title::attr(href)").extract()
        scores = response.css("div.score.unvoted::attr(title)").extract()

        for item in zip(titles, hrefs, scores):

            new_item = RedditItem()

            new_item['title'] = item[0]
            new_item['url'] = item[1]
            new_item['score'] = item[2]

            yield new_item

        # extract link for next page
        next_page = response.css(".next-button").css("a::attr(href)").extract()[0]

        # make request and make this parse become recursive
        yield Request(url=next_page, callback=self.parse)
```

That it, now try to run spider, you will see `Spider` run page to page seem never stop. Have another technical to scrape in multiple page which use `Rule` and `LinkExtractor` , but we will talk about this in another post.

# Resources

| Resource | Link |
| --- | --- |
| CSS selector | https://www.w3schools.com/cssref/css_selectors.asp |
| XPath | https://www.w3schools.com/xml/xpath_intro.asp |