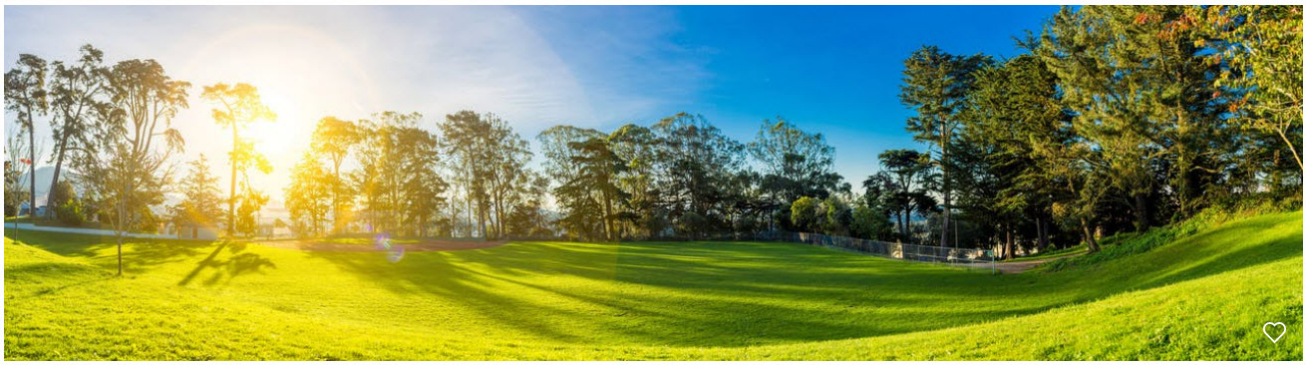
https://tanpham.org

Explain **CrawlSpider**, **LinkExtractor**, **Rule**, **ImagePipeline**

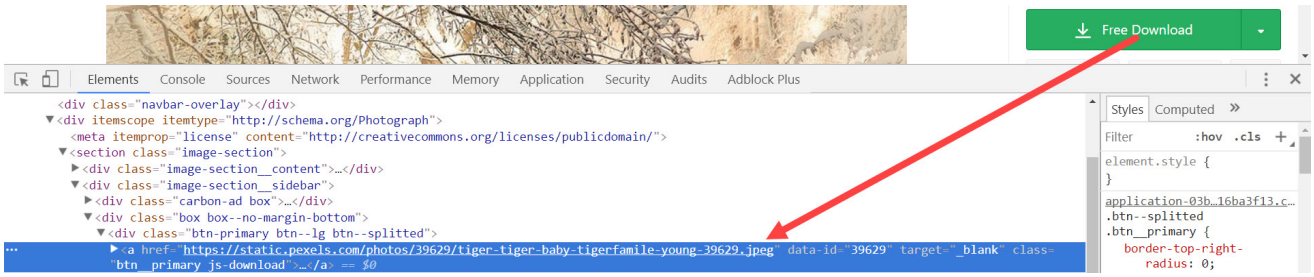
The Objective

This tutorial explain step by step how to scrape nature image from https://www.pexels.com



Understand the Page with Shell

Start from this url which listed all nature picture https://www.pexels.com/search/natural/ , let go to a specific image, our target is url to download image https://www.pexels.com/photo/adult-and-cub-tiger-on-snowfield-near-bare-trees-39629/ . From inspection tool what we need to extract is `href` attribute of `a` tag which have classes are `btn__primary js-download`



Start `shell` with command

```
scrapy shell
```

Fetch the image url

```
fetch('https://www.pexels.com/photo/adult-and-cub-tiger-on-snowfield-near-bare-trees-39629/')
```

Try to css selector and extract image download link

```
response.css('a.btn__primary.js-download::attr(href)').extract()
```

And result is what we need, the full static url to `jpeg` file

```
In [8]: response.css('a.btn__primary.js-download::attr(href)').extract()
Out[8]: [u'https://static.pexels.com/photos/39629/tiger-tiger-baby-tigerfamile-young-39629.jpeg']
```

Define How to Crawl with CrawlSpider

Create a new scrapy project call `pexels` with command

```
scrapy startproject pexels
```

Create a new `crawl` spider name `nature_image` on domain `pexels.com`

```
scrapy genspider -t crawl nature_image pexels.com
```

Let change the `start_urls` to https://www.pexels.com/search/natural/ .And spider file `nature_image.py` look like this

```
# -*- coding: utf-8 -*-
import scrapy
from scrapy.linkextractors import LinkExtractor
from scrapy.spiders import CrawlSpider, Rule


class NatureImageSpider(CrawlSpider):
    name = 'nature_image'
    allowed_domains = ['pexels.com']
    start_urls = ['https://www.pexels.com/search/natural/']

    rules = (
        Rule(LinkExtractor(allow=r'photo/'), callback='parse_item', follow=True),
    )

    def parse_item(self, response):
        i = {}
        #i['domain_id'] = response.xpath('//input[@id="sid"]/@value').extract()
        #i['name'] = response.xpath('//div[@id="name"]').extract()
        #i['description'] = response.xpath('//div[@id="description"]').extract()
        return i
```

Now let's explain what going on in this file:

- `CrawlSpider` : Scrapy provide 2 kind of spider
  - `Spider` is basic one, with this kind of spider we need to care about how to move from page to page by our self
  - `CrawlSpider` provide a mechanism to follow links automatically. Our remain job is specify what kind of link we want to follow. We will use `Rule` and `LinkExtractor` classes for this task
- `Rule` : specify following information:
  - `LinkExtractor` specify what kind of link we want engine to make request. All links are requested need to filtered by regular expression specify by `allow` parameter. In this case, we only do request on link which has `photo/` inside

- `callback` specify function which handle response. In this case is `parse_item` function. We will put parse logic to this function to extract all information we want, in this case is the downloadable image url.
- `follow` we need to specify value to `True` if we want spider follow the link

Let's do some small change with `parse_item` function and make it print out url of detail images

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

Change directory to `pexels` and run spider with command

```
cd pexels
scrapy crawl nature_image
```

The detail image urls show up from console

```
https://www.pexels.com/photo/abandoned-forest-industry-nature-34950/
https://www.pexels.com/photo/daylight-environment-forest-idyllic-459225/
https://www.pexels.com/photo/scenic-view-of-beach-248797/
https://www.pexels.com/photo/wood-light-vacation-picnic-60006/
https://www.pexels.com/photo/close-up-of-hand-holding-plant-against-sky-325702/
https://www.pexels.com/photo/black-pile-of-stones-158607/
```

Now, put the css selector which we already investigate with `shell` to `parse_item` function, then try to print the downloadable image urls

```
def parse_item(self, response):
        print response.css('a.btn__primary.js-download::attr(href)').extract()
```

Start again the crawl and we will see the `jpeg` file from console log.

```
(C:\Users\TAN\Anaconda2) C:\scrapy\pexels>scrapy crawl nature_image
[u'https://static.pexels.com/photos/34950/pexels-photo.jpg']
[u'https://static.pexels.com/photos/459225/pexels-photo-459225.jpeg']
[u'https://static.pexels.com/photos/248797/pexels-photo-248797.jpeg']
[u'https://static.pexels.com/photos/158607/cairn-fog-mystical-background-158607.jpeg']
[u'https://static.pexels.com/photos/132037/pexels-photo-132037.jpeg']
[u'https://static.pexels.com/photos/60006/spring-tree-flowers-meadow-60006.jpeg']
```

Download Image with ImagePipeline

Scrapy provide a way so it is very convenience way to download image.

Step 1 : Enable image pipeline and specify where we want to store images in `settings.py` file

```
ITEM_PIPELINES = {'scrapy.pipelines.images.ImagesPipeline': 1}
IMAGES_STORE = 'images'
```

Step 2 : Edit `items.py` file, add to 2 predefine fields `images` and `image_urls`

```
# -*- 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 PexelsItem(scrapy.Item):
    image_urls = scrapy.Field()
    images = scrapy.Field()
```

Step 3 : Change spider parse function, then `yield` out the item

```
# -*- coding: utf-8 -*-
import scrapy
from scrapy.linkextractors import LinkExtractor
from scrapy.spiders import CrawlSpider, Rule
from pexels.items import PexelsItem


class NatureImageSpider(CrawlSpider):
    # spider name
    name = 'nature_image'

    # only doing request in these domain
    allowed_domains = ['pexels.com']

    # starting points
    start_urls = ['https://www.pexels.com/search/natural/']

    # how to follow the link
    rules = (
        Rule(LinkExtractor(allow=r'photo/'), callback='parse_item', follow=True),
    )

    # parse response
    def parse_item(self, response):
        item = PexelsItem()
        item['image_urls'] = response.css('a.btn__primary.js-download::attr(href)').extract()
        yield item
```

That it, now try out `crawl` command

```
scrapy crawl nature_image
```

And here is result, seem endless stream of image flow from pexels to image folder


1a53bfcc4cca24f5b5bb32ef0ace8d4ebc36bc8e.jpg


1c9ed1face59d89b60291320b58d812bf6a517a8.jpg


1d73c41b24fe5af5b26f6305af9a3efe6945376c.jpg


02e8c5d3d8004989fea1fd0a3d661f44eb2f744a.jpg


3ab5ad8223def4138fb3c0adb90d82b41e101742.jpg


4eb379cd4a179aa33b1f1123ddedf6368481ef8c.jpg


5cc4411db3b3e5a13e717c1329030ee8560ee1e5.jpg


8a412c76aaddd15365d1c4a26deb72676cdcfa63.jpg


8c4c67daacb287bb9eaace61b7379bbc8009dfbc.jpg


8cc4b07d1d43476a0ebd82e9f8555361eec20418.jpg


9a38e78c71e61dd0ab8037ab20cb55bfcfa155ec.jpg


9c7d06553c7364ee0cf0d013578170e9b521970c.jpg


9e1bb9b8656f7b2c26738352bd4f0839669906d9.jpg


25ee6b916ce9ba176773166f048452973dea112b.jpg


35d5d64e06f95e08777f276f719f1b24c4bfdfde.jpg


65a4c0a9d5117219005bc621a70f3da28573d057.jpg