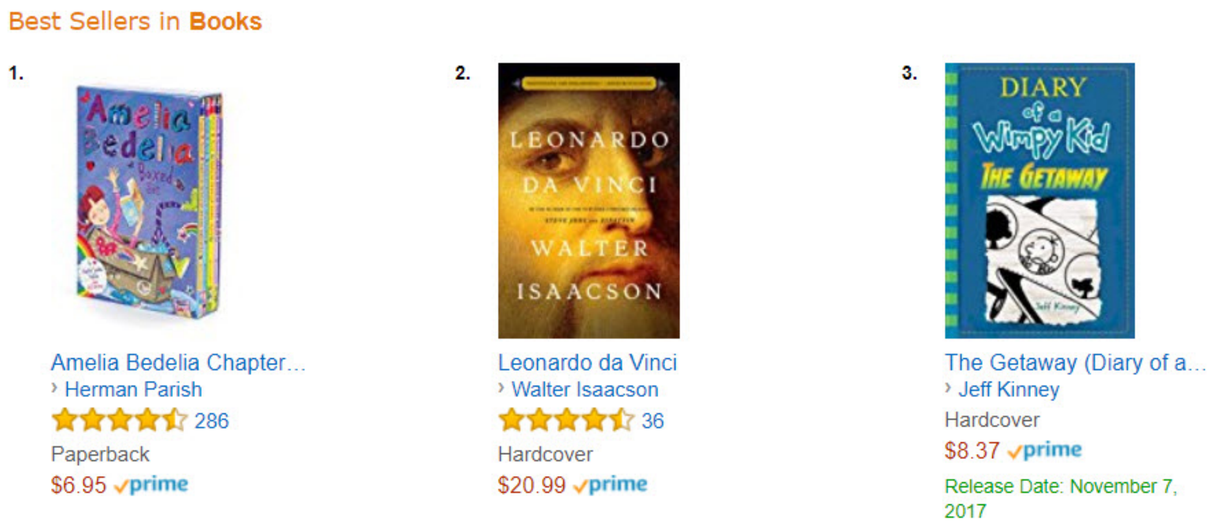
Explain **Inspection Tool**, **Multiple Level Parse function**.

The Objective

In this tutorial we will scape data from Amazon. Open link on browser https://www.amazon.com/best-sellers-books-Amazon/zgbs/books the page contain 100 best seller book will show up.



Best Sellers in Books

1. Amelia Bedelia Chapter…
› Herman Parish
★★★★☆ 286
Paperback
$6.95 ✓prime

2. Leonardo da Vinci
› Walter Isaacson
★★★★☆ 36
Hardcover
$20.99 ✓prime

3. The Getaway (Diary of a…
› Jeff Kinney
Hardcover
$8.37 ✓prime
Release Date: November 7, 2017

The objective of this tutorial will be scrape following data item for each book and then save data to a csv file

- Sell order
- Book title
- Author

Understand starting page

As every other scraping data project, first step should be about understand the page, how to extract it.

Open the `shell` with command

```
scrapy shell
```

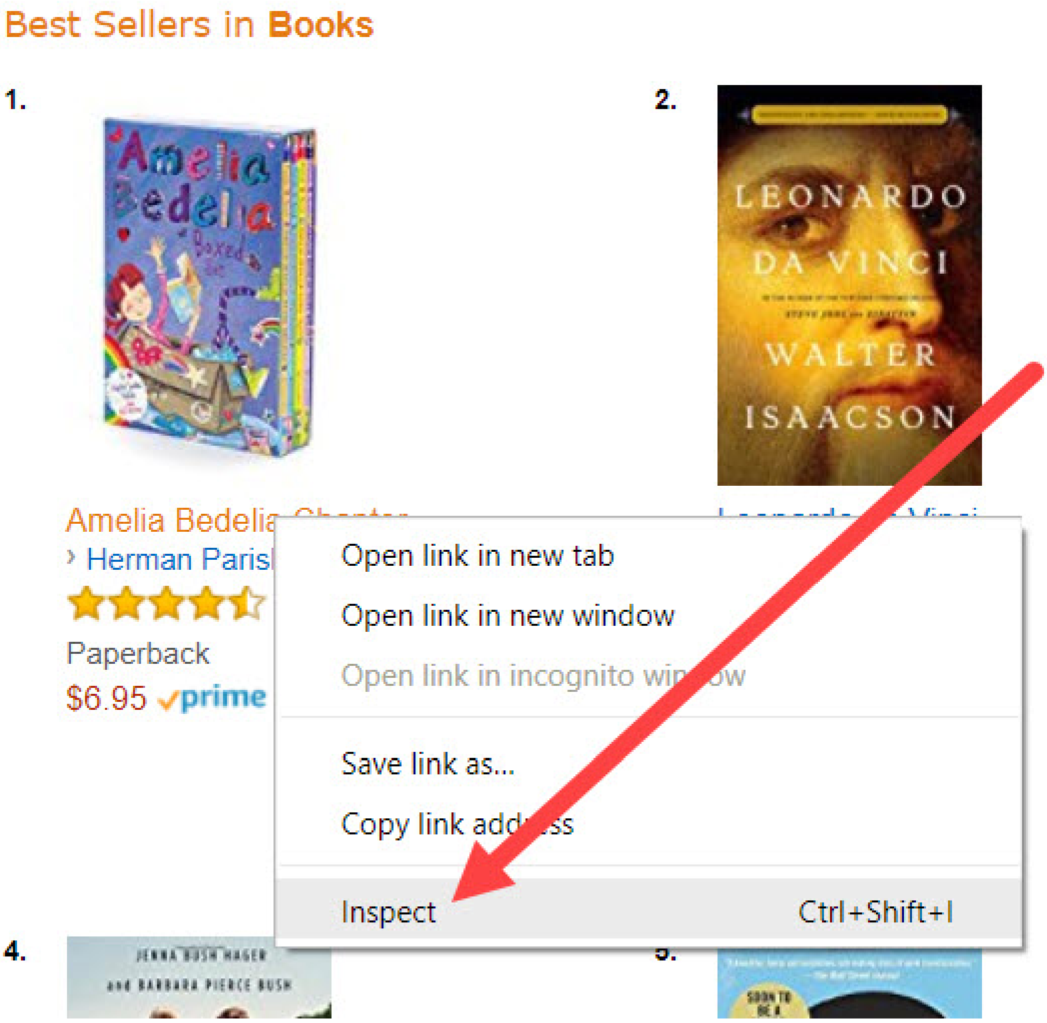
From the `shell` , fetch the starting page with command

```
fetch('https://www.amazon.com/best-sellers-books-Amazon/zgbs/books')
```

After `fetch` command, a `response` object is created. Let try to view the html source from `response` with command

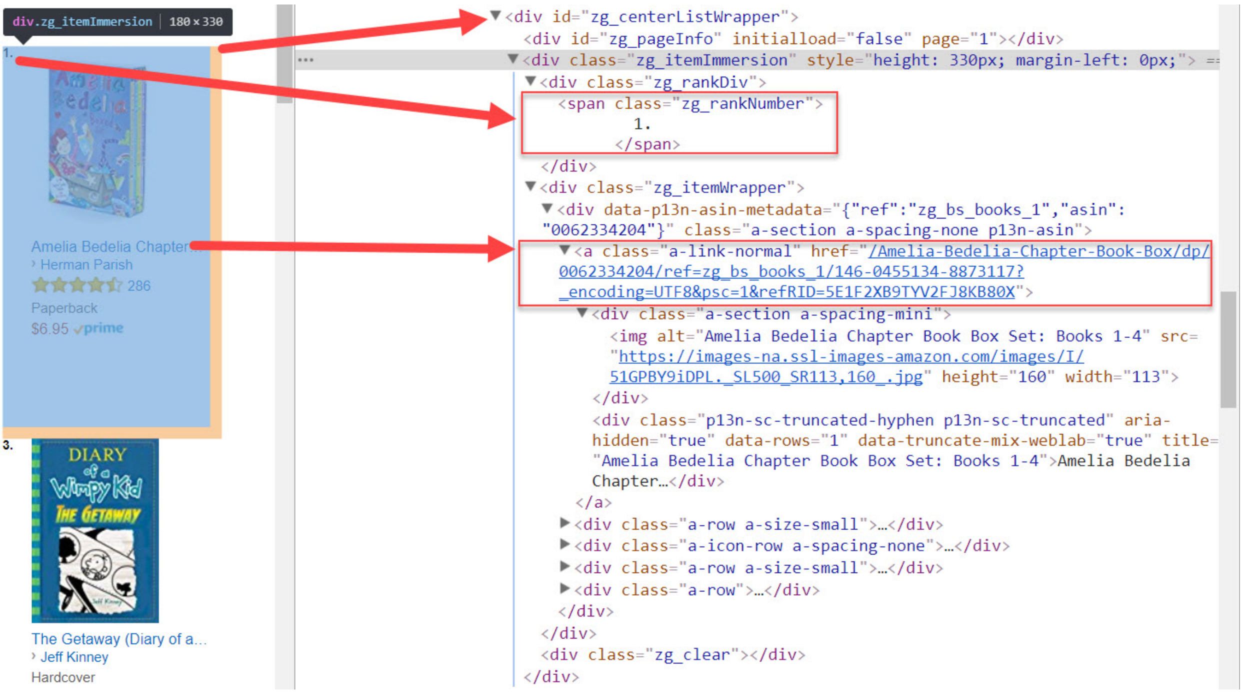
```
response.text
```

Now let's inspect the page with developer tool from Chrome browser. Move mouse over one of the title, then right click, chose `inspect`.



From inspection, We could see strategy to extract information:

- Each book block is inside a `div` with class name `zg_itemImmersion`
- Inside father block the rank could be get by `span` tag with class name `zg_rankNumber`
- Inside father block the link to detail book information could be get by `a` tag with class name `a-link-normal`



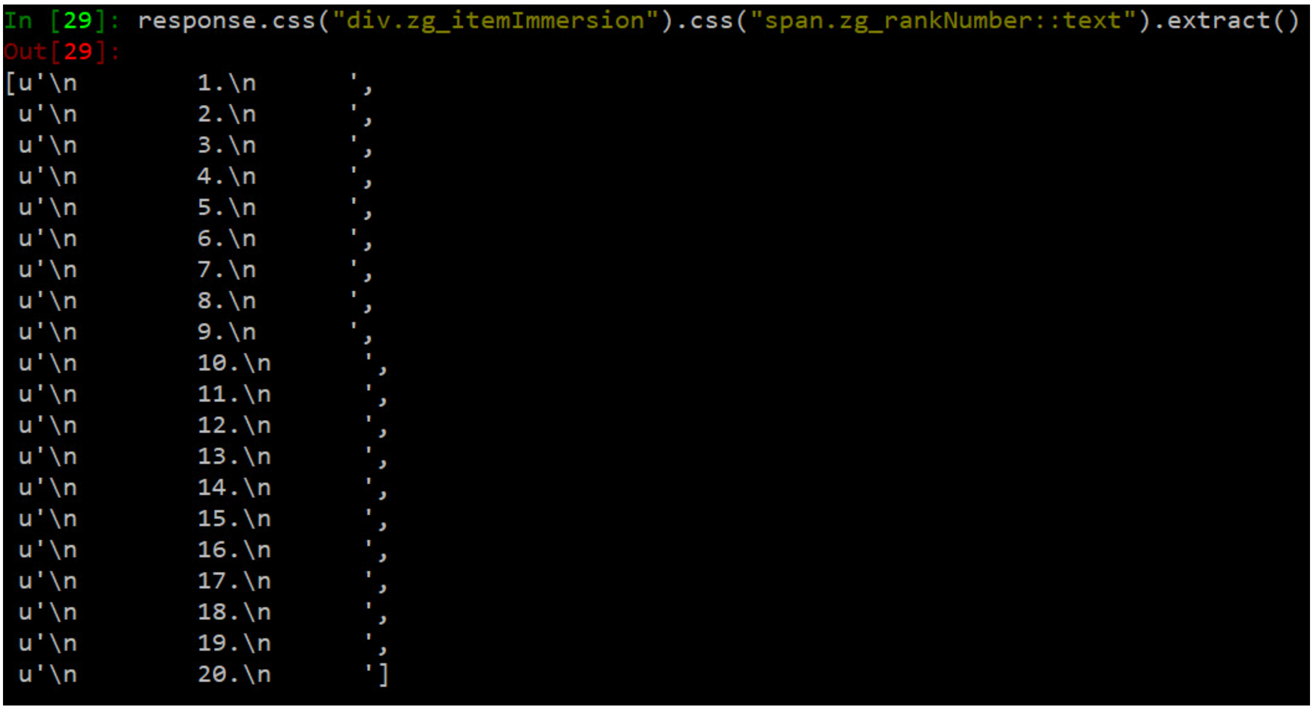
div.zg_itemImmersion | 180 × 330
1.
Amelia Bedelia Chapter...
› Herman Parish
286
Paperback
$6.95 ✓prime
3.
DIARY of a Wimpy Kid THE GETAWAY
The Getaway (Diary of a...
› Jeff Kinney
Hardcover

```
▼<div id="zg_centerListWrapper">
   <div id="zg_pageInfo" initialload="false" page="1"></div>
 ▼<div class="zg_itemImmersion" style="height: 330px; margin-left: 0px;"> ==
   ▼<div class="zg_rankDiv">
      <span class="zg_rankNumber">
              1.
            </span>
    </div>
   ▼<div class="zg_itemWrapper">
     ▼<div data-p13n-asin-metadata="{"ref":"zg_bs_books_1","asin":
      "0062334204"}" class="a-section a-spacing-none p13n-asin">
       ▼<a class="a-link-normal" href="/Amelia-Bedelia-Chapter-Book-Box/dp/
        0062334204/ref=zg_bs_books_1/146-0455134-8873117?
        _encoding=UTF8&psc=1&refRID=5E1F2XB9TYV2FJ8KB80X">
         ▼<div class="a-section a-spacing-mini">
            <img alt="Amelia Bedelia Chapter Book Box Set: Books 1-4" src=
            "https://images-na.ssl-images-amazon.com/images/I/
            51GPBY9iDPL._SL500_SR113,160_.jpg" height="160" width="113">
          </div>
          <div class="p13n-sc-truncated-hyphen p13n-sc-truncated" aria-
          hidden="true" data-rows="1" data-truncate-mix-weblab="true" title=
          "Amelia Bedelia Chapter Book Box Set: Books 1-4">Amelia Bedelia
          Chapter…</div>
        </a>
       ▶<div class="a-row a-size-small">…</div>
       ▶<div class="a-icon-row a-spacing-none">…</div>
       ▶<div class="a-row a-size-small">…</div>
       ▶<div class="a-row">…</div>
      </div>
    </div>
    <div class="zg_clear"></div>
  </div>
```

Now try to extract the rank with css selector and `response` object

```
response.css("div.zg_itemImmersion").css("span.zg_rankNumber::text").extract()
```

Rank show up

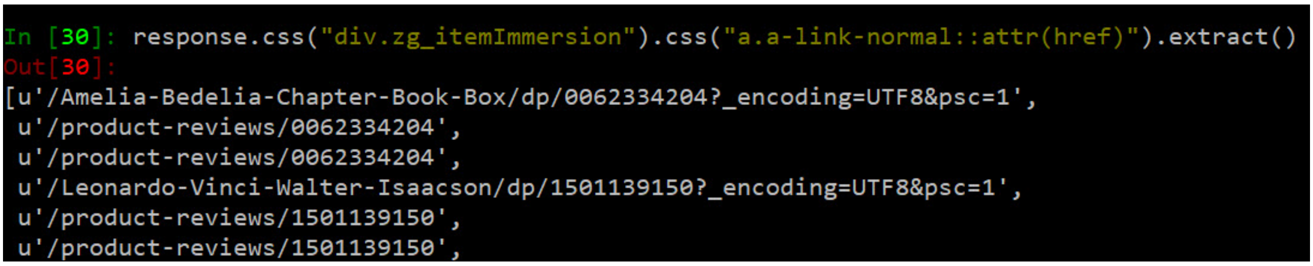


```
In [29]: response.css("div.zg_itemImmersion").css("span.zg_rankNumber::text").extract()
Out[29]:
[u'\n            1.\n          ',
 u'\n            2.\n          ',
 u'\n            3.\n          ',
 u'\n            4.\n          ',
 u'\n            5.\n          ',
 u'\n            6.\n          ',
 u'\n            7.\n          ',
 u'\n            8.\n          ',
 u'\n            9.\n          ',
 u'\n            10.\n          ',
 u'\n            11.\n          ',
 u'\n            12.\n          ',
 u'\n            13.\n          ',
 u'\n            14.\n          ',
 u'\n            15.\n          ',
 u'\n            16.\n          ',
 u'\n            17.\n          ',
 u'\n            18.\n          ',
 u'\n            19.\n          ',
 u'\n            20.\n          ']
```

Let try to extract the detail book link

```
response.css("div.zg_itemImmersion").css("a.a-link-normal::attr(href)").extract()
```

Return not just have detail link but also has product link, so we need to do filter when coding the `spider`



```
In [30]: response.css("div.zg_itemImmersion").css("a.a-link-normal::attr(href)").extract()
Out[30]:
[u'/Amelia-Bedelia-Chapter-Book-Box/dp/0062334204?_encoding=UTF8&psc=1',
 u'/product-reviews/0062334204',
 u'/product-reviews/0062334204',
 u'/Leonardo-Vinci-Walter-Isaacson/dp/1501139150?_encoding=UTF8&psc=1',
 u'/product-reviews/1501139150',
 u'/product-reviews/1501139150',
```

That quite enough of start page understanding, let do project and spider.

Get Detail Book Links

Create a new project call **amazon**

```
scrapy startproject amazon
```

Change current directory to amazon folder and create `spider` call **book**

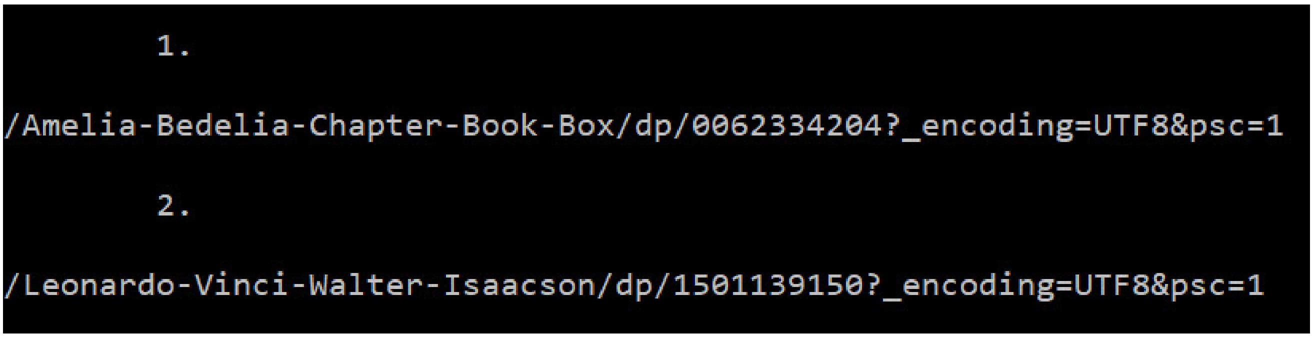
```
scrapy genspider book www.amazon.com
```

Let change `start_urls` and `parse` function and you have some thing like this

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


class BookSpider(scrapy.Spider):

    name = 'book'
    allowed_domains = ['www.amazon.com']
    start_urls = ['https://www.amazon.com/best-sellers-books-Amazon/zgbs/books']

    def parse(self, response):

        # extract data from response
        ranks = response.css("div.zg_itemImmersion").css("span.zg_rankNumber::text").extract()
        links = response.css("div.zg_itemImmersion").css("a.a-link-
normal::attr(href)").extract()

        detail_links = []

        # filter product reviews link
        for link in links:
            if 'product-reviews' not in link:
                detail_links.append(link)

        for item in zip(ranks, detail_links):
            print item[0]
            print item[1]
```

Now try to run this `spider` with `crawl` command and you will get rank and detail link print out

```
scrapy crawl book
```



```
        1.
/Amelia-Bedelia-Chapter-Book-Box/dp/0062334204?_encoding=UTF8&psc=1
        2.
/Leonardo-Vinci-Walter-Isaacson/dp/1501139150?_encoding=UTF8&psc=1
```

Define Scrape Data with Item

It is time to define data structure which we want to archive inside file `items.py` . Change file `items.py` as follow:

```
# -*- 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 AmazonItem(scrapy.Item):
    order = scrapy.Field()
    title = scrapy.Field()
    author = scrapy.Field()
```

Multiple Levels Parsing

Back to spider file, from Amazon page structure. We see that, to get data we should have 2 levels of extraction.

- First extraction happen at starting url, and we could extract `order` and `detail link`
- From `detail link` , we continue to do another `request` , get the `response` and parse remain information : title, author, price, summary and cover image by another function call `parse_detail_info`

Now let change the `parse` function follow changing in file `items.py`

```
# -*- coding: utf-8 -*-
import scrapy
# import the item
from amazon.items import AmazonItem


class BookSpider(scrapy.Spider):

    name = 'book'
    allowed_domains = ['www.amazon.com']
    start_urls = ['https://www.amazon.com/best-sellers-books-Amazon/zgbs/books']

    def parse(self, response):

        # extract data from response
        orders = response.css("div.zg_itemImmersion").css("span.zg_rankNumber::text").extract()
        links = response.css("div.zg_itemImmersion").css("a.a-link-normal::attr(href)").extract()

        detail_links = []

        # filter product reviews out
        for link in links:
            if 'product-reviews' not in link:
                detail_links.append('https://www.amazon.com/'+link)

        # create data item
        for item in zip(orders, detail_links):
            # create a new item
            new_item = AmazonItem()
            new_item['order'] = item[0]

            # # create a new request and get detail infor on parse detail
            request = scrapy.Request(url=item[1], callback=self.parse_detail_info)

            # transfer item to parse detail function
            request.meta['item'] = new_item

            yield request


    def parse_detail_info(self, response):
        item = response.meta['item']
        print item['order']
        print response.url
```

Note that with Amazon you need to specify the user agent inside `settings.py` file

```
# Need to have when want to crawl from Amazon
USER_AGENT = 'amazon (+https://tanpham.org)'
```

Now start the crawl and you will see function `parse_detail_info` work as expected.

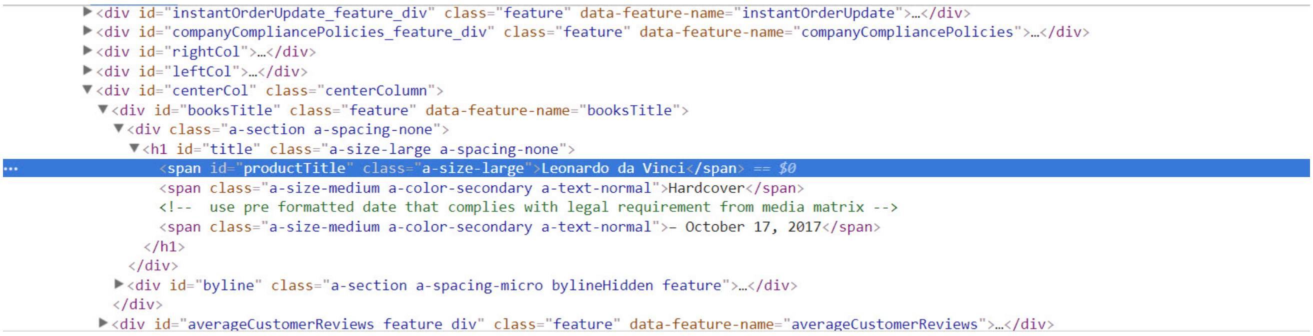
Scrape Book Title, Author, Intro

Let's go back to `shell` to see how to scrape data from detail book page. Try with one detail book page.

```
scrapy shell
```

```
fetch('https://www.amazon.com/Leonardo-Vinci-Walter-Isaacson/dp/1501139150/ref=zg_bs_books_1?
_encoding=UTF8&psc=1&refRID=7TPBA5D9KY75PJ6S8YHT')
```

Now do some inspection with Chrome developer tool. First item is book title



```
▶<div id="instantOrderUpdate_feature_div" class="feature" data-feature-name="instantOrderUpdate">…</div>
▶<div id="companyCompliancePolicies_feature_div" class="feature" data-feature-name="companyCompliancePolicies">…</div>
▶<div id="rightCol">…</div>
▶<div id="leftCol">…</div>
▼<div id="centerCol" class="centerColumn">
  ▼<div id="booksTitle" class="feature" data-feature-name="booksTitle">
    ▼<div class="a-section a-spacing-none">
      ▼<h1 id="title" class="a-size-large a-spacing-none">
...       <span id="productTitle" class="a-size-large">Leonardo da Vinci</span> == $0
          <span class="a-size-medium a-color-secondary a-text-normal">Hardcover</span>
          <!--  use pre formatted date that complies with legal requirement from media matrix -->
          <span class="a-size-medium a-color-secondary a-text-normal">– October 17, 2017</span>
        </h1>
      </div>
    ▶<div id="byline" class="a-section a-spacing-micro bylineHidden feature">…</div>
    </div>
  ▶<div id="averageCustomerReviews_feature_div" class="feature" data-feature-name="averageCustomerReviews">…</div>
```

For book title, we could extract by finding `span` tag with `id=productTitle`

```
response.css('span#productTitle::text').extract()
```

For author name

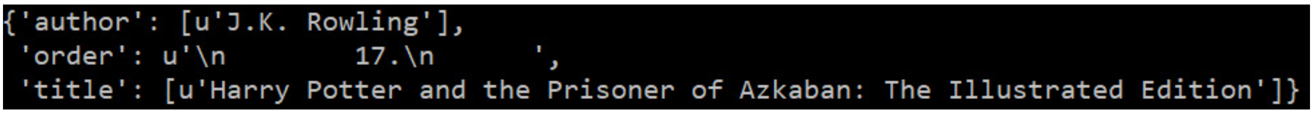
```
response.css('a.contributorNameID::text').extract()
```

Now let modify the function `parse_detail_info` in file `book.py` so we will scrape book title and author name.

```
def parse_detail_info(self, response):

        item = response.meta['item']
        item['title'] = response.css('span#productTitle::text').extract()
        item['author'] = response.css('a.contributorNameID::text').extract()

        yield item
```

That it, let try `crawl` command to see item printed out

```
cd amazon
scrapy crawl book
```

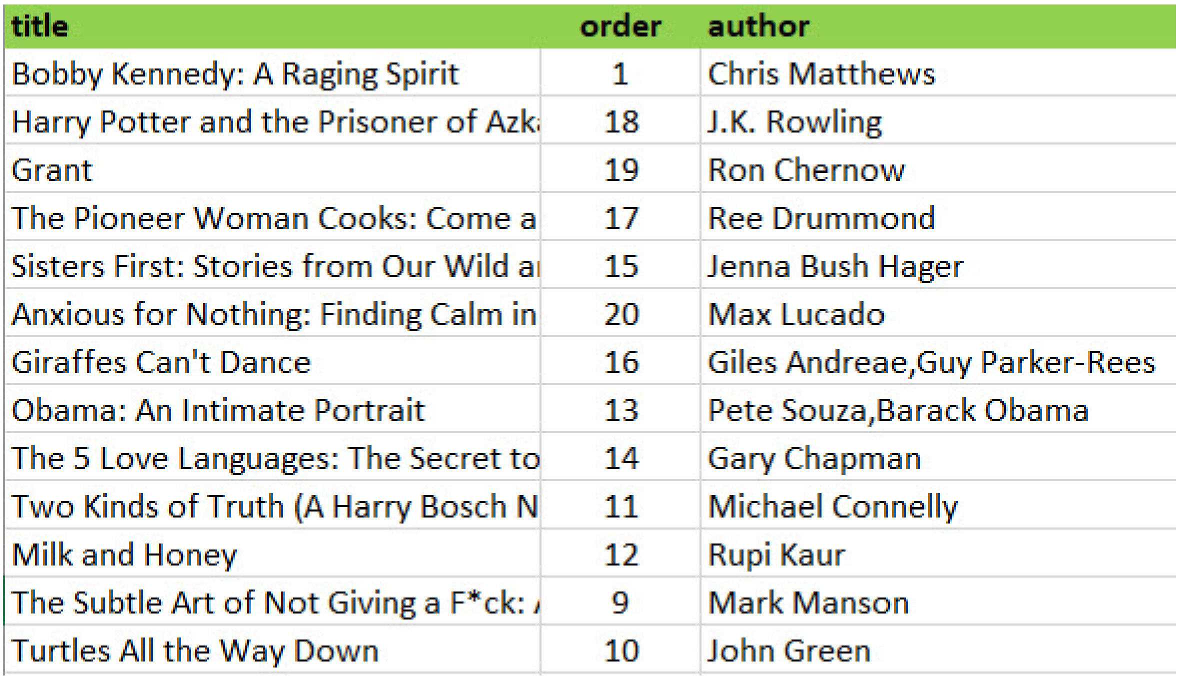


```
{'author': [u'J.K. Rowling'],
 'order': u'\n          17.\n          ',
 'title': [u'Harry Potter and the Prisoner of Azkaban: The Illustrated Edition']}
```

Let's save data to csv with command

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

And we have data



title	order	author
Bobby Kennedy: A Raging Spirit	1	Chris Matthews
Harry Potter and the Prisoner of Azk	18	J.K. Rowling
Grant	19	Ron Chernow
The Pioneer Woman Cooks: Come a	17	Ree Drummond
Sisters First: Stories from Our Wild a	15	Jenna Bush Hager
Anxious for Nothing: Finding Calm in	20	Max Lucado
Giraffes Can't Dance	16	Giles Andreae,Guy Parker-Rees
Obama: An Intimate Portrait	13	Pete Souza,Barack Obama
The 5 Love Languages: The Secret to	14	Gary Chapman
Two Kinds of Truth (A Harry Bosch N	11	Michael Connelly
Milk and Honey	12	Rupi Kaur
The Subtle Art of Not Giving a F*ck:	9	Mark Manson
Turtles All the Way Down	10	John Green