راهنمای استخراج اطلاعات از وب

بیایید شما را با استخراج اطلاعات از وب و پایتون آشنا کنیم. قبل از شروع، برخی از قوانین مهمی که باید رعایت و درک کنید را معرفی می‌کنیم:

1. همیشه محترمانه عمل کنید و سعی کنید اجازه استخراج اطلاعات را دریافت کنید. از سایت‌ها درخواست‌های استخراج اطلاعات بدون مجوز نکنید، در غیر اینصورت آدرس IP شما ممکن است مسدود شود!
2. به خاطر داشته باشید که وبسایت‌ها به طور مداوم تغییر می‌کنند، به این معنی که کد شما ممکن است از کار بیافتد و کاملاً خراب شود.
3. تقریباً هر پروژهٔ استخراج اطلاعات مورد علاقه، یک پروژهٔ منحصربه‌فرد و سفارشی است، بنابراین تلاش کنید تا مهارت‌های یادگرفته شده را به صورت کلی تعمیم دهید.

خوب، بیایید با مباحث پایه شروع کنیم!

اجزای پایه یک وب‌سایت

HTML

HTML مخفف عبارت Hypertext Markup Language است و هر وب‌سایتی در اینترنت از آن برای نمایش اطلاعات استفاده می‌کند. حتی سیستم دفترچه یادداشت جوپیتر از آن برای نمایش این اطلاعات در مرورگر شما استفاده می‌کند. اگر روی یک وب‌سایت راست کلیک کرده و گزینه "View Page Source" را انتخاب کنید، می‌توانید HTML خام یک صفحه وب را ببینید. این اطلاعاتی هستند که پایتون برای استخراج اطلاعات از آن‌ها استفاده می‌کند. بیایید به یک نمونه ساده از HTML یک صفحه وب نگاهی بیندازیم:

```
<!DOCTYPE html>
<html>
    <head>
        <title>عنوان تب مرورگر</title>
    </head>
    <body>
        <h1>هدر وبسایت</h1>
        <p>یک پاراگراف</p>
    <body>
</html>
```

بیایید این اجزا را تجزیه کنیم.

هر برچسب `<tag>` نوع مشخصی از بلوک را در صفحه وب نشان می‌دهد:

1. <!DOCTYPE html> سند HTML همیشه با این اعلام نوع آغاز می‌شود تا مرورگر بداند که این یک فایل HTMLاست.
2. بلوک‌های اجزای سند HTML بین <html> و </html> قرار می‌گیرند.
3. متادیتا و اتصالات اسکریپت) مانند لینک به یک فایل CSS یا یک فایل (JS معمولاً در بلوک <head> قرار می‌گیرند.
4. بلوک برچسب <title> عنوان صفحه وب را تعریف می‌کند (که در تب یک وب‌سایتی که شما در آن قرار دارید نمایش داده می‌شود).
5. بین برچسب‌های <body> و </body> بلوک‌هایی قرار می‌گیرند که قابل مشاهده برای بازدیدکنندگان سایت

CSS

CSS مخفف عبارت Cascading Style Sheets است، این است که "استایل" را به یک وب‌سایت می‌دهد، از جمله رنگ‌ها و قلم‌ها و حتی برخی از انیمیشن‌ها! CSS از برچسب‌های مانند **id** یا **class** برای اتصال یک

عنصر HTML به یک ویژگی CSS استفاده می‌کند، مانند رنگ خاصی. **id** یک شناسه منحصر به فرد برای برچسب HTML است و باید در سند HTML منحصراً یکتا باشد، به طور کلی یک اتصال یک‌باره است. **class** یک استایل عمومی را تعریف می‌کند که سپس می‌تواند به چندین برچسب HTML مرتبط شود. در واقع، اگر می‌خواهید فقط یک برچسب HTML را قرمز کنید، از برچسب id استفاده خواهید کرد و اگر می‌خواهید چندین برچسب/بلوک HTML را قرمز کنید، یک کلاس در سند CSS خود ایجاد کرده و سپس آن را به بقیه این بلوک‌ها متصل خواهید کرد.

راهنمایی‌های Scraping

در نظر داشته باشید که همیشه برای وب‌سایتی که قرار است اطلاعات آن را استخراج کنید، باید اجازه بگیرید! شرایط و ضوابط یک وب‌سایت را بررسی کنید تا اطلاعات بیشتری در این زمینه بدست آورید. همچنین به یاد داشته باشید که یک کامپیوتر می‌تواند درخواست‌ها را به سرعت بسیار زیادی به یک وب‌سایت ارسال کند، بنابراین اگر درخواست‌های زیادی را به سرعت بیش از حد ارسال کنید، وب‌سایت ممکن است آدرس IP کامپیوتر شما را مسدود کند. در نهایت، وب‌سایت‌ها به طور مداوم تغییر می‌کنند! احتمالاً برای وظایف استخراج اطلاعات بلندمدت، باید کد خود را به طور مکرر به‌روز رسانی کنید.

استخراج اطلاعات از وب با استفاده از Python

چندین کتابخانه نیاز خواهید داشت. می‌توانید به خط فرمان خود بروید و آنها را با استفاده از دستور conda install (اگر از توزیع Anaconda استفاده می‌کنید) یا pip install برای سایر توزیع‌های پایتون نصب کنید.

```
conda install requests
conda install lxml
conda install bs4
```

اگر از نصب Anaconda استفاده نمی‌کنید، می‌توانید به جای **conda install** از **pip install** استفاده کنید. به عنوان مثال:

```
pip install requests
pip install lxml
pip install bs4
```

حالا بیایید ببینیم با این کتابخانه‌ها چه کارهایی می‌توانیم انجام دهیم.

تسک نمونه ۰ - دریافت عنوان یک صفحه

بیایید از یک مثال بسیار ساده شروع کنیم و عنوان یک صفحه را دریافت کنیم. به یاد داشته باشید که این بلوک HTML با برچسب **title** است. برای این تسک ما از www.example.com (http://www.example.com) استفاده می‌کنیم که یک وب‌سایت به صورت ویژه برای استفاده به عنوان دامنه نمونه ساخته شده است. بیایید از مراحل اصلی عبور کنیم:

In [1]:

```
import requests
```

In [2]:

```
# Step 1: Use the requests library to grab the page
# Note, this may fail if you have a firewall blocking Python/Jupyter
# Note sometimes you need to run this twice if it fails the first time
res = requests.get("http://www.example.com")
```

این شیء یک شیء requests.models.Response است و در واقع حاوی اطلاعات وب‌سایت است، به عنوان مثال:

In [3]:

```
type(res)
```

Out[3]:

```
requests.models.Response
```

In [4]:

```
res.text
```

Out[4]:

```
'<!doctype html>\n<html>\n<head>\n    <title>Example Domain</title>\n\n
<meta charset="utf-8" />\n    <meta http-equiv="Content-type" content="tex
t/html; charset=utf-8" />\n    <meta name="viewport" content="width=device
-width, initial-scale=1" />\n    <style type="text/css">\n    body {\n
background-color: #f0f0f2;\n        margin: 0;\n        padding: 0;\n
font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Op
en Sans", "Helvetica Neue", Helvetica, Arial, sans-serif;\n        \n    }
\n    div {\n        width: 600px;\n        margin: 5em auto;\n        pad
ding: 2em;\n        background-color: #fdfdff;\n        border-radius: 0.5
em;\n        box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);\n    }\n    a:l
ink, a:visited {\n        color: #38488f;\n        text-decoration: non
e;\n    }\n    @media (max-width: 700px) {\n        div {\n            mar
gin: 0 auto;\n            width: auto;\n        }\n    }\n    </style>
\n</head>\n\n<body>\n<div>\n    <h1>Example Domain</h1>\n    <p>This domai
n is for use in illustrative examples in documents. You may use this\n
domain in literature without prior coordination or asking for permission.
</p>\n    <p><a href="https://www.iana.org/domains/example">More informati
on...</a></p>\n</div>\n</body>\n</html>\n'
```

حالا از BeautifulSoup برای تجزیه و تحلیل صفحه استخراج شده استفاده می‌کنیم. در واقع می‌توانستیم از اسکریپت سفارشی خودمان برای جستجوی موارد در رشته **res.text** استفاده کنیم، اما کتابخانه BeautifulSoup قابلیت‌ها و روش‌های زیادی برای دریافت اطلاعات از یک رشته از این نوع (به طور کلی یک فایل HTML) دارد. با استفاده از BeautifulSoup می‌توانیم یک شیء "soup" ایجاد کنیم که حاوی "مواد" صفحه وب است. نپرسید من اسامی عجیب کتابخانه‌ها را انتخاب

In [5]:

```
1 import bs4
```

In [6]:

```
1 soup = bs4.BeautifulSoup(res.text,"lxml")
```

In [7]:

```
1 soup
```

Out[7]:

```
<!DOCTYPE html>
<html>
<head>
<title>Example Domain</title>
<meta charset="utf-8"/>
<meta content="text/html; charset=utf-8" http-equiv="Content-type"/>
<meta content="width=device-width, initial-scale=1" name="viewport"/>
<style type="text/css">
    body {
        background-color: #f0f0f2;
        margin: 0;
        padding: 0;
        font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe
UI", "Open Sans", "Helvetica Neue", Helvetica, Arial, sans-serif;

    }
    div {
        width: 600px;
        margin: 5em auto;
        padding: 2em;
        background-color: #fdfdff;
        border-radius: 0.5em;
        box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
    }
    a:link, a:visited {
        color: #38488f;
        text-decoration: none;
    }
    @media (max-width: 700px) {
        div {
            margin: 0 auto;
            width: auto;
        }
    }
    </style>
</head>
<body>
<div>
<h1>Example Domain</h1>
<p>This domain is for use in illustrative examples in documents. You may u
se this
    domain in literature without prior coordination or asking for permissi
on.</p>
<p><a href="https://www.iana.org/domains/example">More information...</a>
</p>
</div>
</body>
</html>
```

حالا بیایید از متد `.select()` برای دریافت عناصر استفاده کنیم. ما به دنبال برچسب 'title' هستیم، بنابراین عبارت 'title' را به عنوان ورودی ارسال می‌کنیم.

In [8]:

```
soup.select('title')
```

Out[8]:

```
[<title>Example Domain</title>]
```

توجه کنید که چه چیزی در اینجا برگشت داده می‌شود، در واقع یک لیست است که حاوی تمام عناصر عنوان (با برچسب‌هایشان) است. شما می‌توانید با استفاده از ایندکس‌گذاری یا حتی حلقه‌بازیابی عناصر را از لیست دریافت کنید. از آنجا که این شیء هنوز یک برچسب ویژه است، می‌توانیم از فراخوانی‌های متد برای دریافت متن فقط استفاده کنیم.

In [9]:

```
title_tag = soup.select('title')
```

In [10]:

```
title_tag[0]
```

Out[10]:

```
<title>Example Domain</title>
```

In [11]:

```
type(title_tag[0])
```

Out[11]:

```
bs4.element.Tag
```

In [12]:

```
title_tag[0].getText()
```

Out[12]:

```
'Example Domain'
```

تسک نمونه ۱ - دریافت همه عناصر یک کلاس

بیایید سعی کنیم تمام عناوین بخش‌های مقاله ویکی‌پدیا درباره Grace Hopper را از این URL دریافت کنیم:
https://en.wikipedia.org/wiki/Grace_Hopper (https://en.wikipedia.org/wiki/Grace_Hopper)

In [13]:

```
# First get the request
res = requests.get('https://en.wikipedia.org/wiki/Grace_Hopper')
```

In [14]:

```
# Create a soup from request
soup = bs4.BeautifulSoup(res.text,"lxml")
```

حالا وقت آن رسیده است که بفهمیم به دنبال چه چیزی هستیم. عنصر را در صفحه بررسی کنید و ببینید که عناوین بخش‌ها دارای کلاس "mw-headline" هستند. از آنجا که این یک کلاس است و نه یک برچسب ساده، باید به برخی از نحوه نوشتن CSS پایبند باشیم. در این حالت:

Match Results	Syntax to pass to the .select() method
All elements with the `<div>` tag	`soup.select('div')`
The HTML element containing the `id` attribute of `some_id`	`soup.select('#some_id')`
All the HTML elements with the CSS `class` named `notice`	`soup.select('.notice')`
<Any elements named `<span>` that are within an element named <div	`soup.select('div span')`
Any elements named `<span>` that are *directly* within an element named `<div>`, with no other element in between	`soup.select('div > span')`

In [15]:

```
# note depending on your IP Address,
# this class may be called something different
soup.select(".mw-parser-output")
```

```
 <li><a class="mw-redirect" href="/wiki/Enterprise_Unified_Process" tit
le="Enterprise Unified Process">EUP</a></li>
 <li><a href="/wiki/Executable_UML" title="Executable UML">Executable U
ML</a></li>
 <li><a href="/wiki/Incremental_build_model" title="Incremental build m
odel">Incremental model</a></li>
 <li><a href="/wiki/Iterative_and_incremental_development" title="Itera
tive and incremental development">Iterative model</a></li>
 <li><a href="/wiki/Software_prototyping" title="Software prototyping">
Prototype model</a></li>
 <li><a href="/wiki/Rapid_application_development" title="Rapid applica
tion development">RAD</a></li>
 <li><a href="/wiki/Unified_Process" title="Unified Process">UP</a></li
>
 <li><a href="/wiki/Scrum_(software_development)" title="Scrum (softwar
e development)">Scrum</a></li>
 <li><a href="/wiki/Spiral_model" title="Spiral model">Spiral model</a>
</li>
 <li><a class="mw-redirect" href="/wiki/V-Model_(software_development)"
title="V-Model (software development)">V-Model</a></li>
```

In [17]:

```
for item in soup.select(".mw-parser-output"):
    print("**********************************")
    print(item.text)
```

```
Mary Engle Pennington
Mercy Otis Warren
2003
Linda G. Alvarado
Donna de Varona
Gertrude Ederle
Martha Matilda Harper
Patricia Roberts Harris
Stephanie L. Kwolek
Dorothea Lange
Mildred Robbins Leet
Patsy Takemoto Mink
Sacagawea
Anne Sullivan
Sheila E. Widnall
2005
Florence E. Allen
Ruth Fulton Benedict
Betty Bumpers
Hillary Clinton
```

تسک نمونه ۳ - دریافت تصویر از یک وبسایت

بیایید سعی کنیم تصویر رایانه دیپ بلو را از این مقاله ویکی‌پدیا دریافت کنیم:
https://en.wikipedia.org/wiki/Deep_Blue_(chess_computer)
(https://en.wikipedia.org/wiki/Deep_Blue_(chess_computer))

In [18]:

```
res = requests.get("https://en.wikipedia.org/wiki/Deep_Blue_(chess_computer)")
```

In [19]:

```
soup = bs4.BeautifulSoup(res.text,'lxml')
```

In [20]:

```
image_info = soup.select('.thumbimage')
```

In [21]:

```
1 image_info
```

Out[21]:

```
[<img class="thumbimage" data-file-height="600" data-file-width="800" deco
ding="async" height="165" src="//upload.wikimedia.org/wikipedia/commons/th
umb/6/6f/Kasparov_Magath_1985_Hamburg-2.png/220px-Kasparov_Magath_1985_Ham
burg-2.png" srcset="//upload.wikimedia.org/wikipedia/commons/thumb/6/6f/Ka
sparov_Magath_1985_Hamburg-2.png/330px-Kasparov_Magath_1985_Hamburg-2.png
1.5x, //upload.wikimedia.org/wikipedia/commons/thumb/6/6f/Kasparov_Magath_
1985_Hamburg-2.png/440px-Kasparov_Magath_1985_Hamburg-2.png 2x" width="22
0"/>,
 <img class="thumbimage" data-file-height="2756" data-file-width="2067" de
coding="async" height="293" src="//upload.wikimedia.org/wikipedia/commons/
thumb/8/83/One_of_Deep_Blue%27s_processors_%282586060990%29.jpg/220px-One_
of_Deep_Blue%27s_processors_%282586060990%29.jpg" srcset="//upload.wikimed
ia.org/wikipedia/commons/thumb/8/83/One_of_Deep_Blue%27s_processors_%28258
6060990%29.jpg/330px-One_of_Deep_Blue%27s_processors_%282586060990%29.jpg
1.5x, //upload.wikimedia.org/wikipedia/commons/thumb/8/83/One_of_Deep_Blu
e%27s_processors_%282586060990%29.jpg/440px-One_of_Deep_Blue%27s_processor
s_%282586060990%29.jpg 2x" width="220"/>]
```

In [22]:

```
1 len(image_info)
```

Out[22]:

```
2
```

In [23]:

```
1 computer = image_info[0]
2
```

In [24]:

```
1 type(computer)
```

Out[24]:

```
bs4.element.Tag
```

شما می‌توانید برای بخش‌هایی از برچسب تماس‌های دیکشنری‌مانند صدا زده شوید، در این حالت، ما به **src** یا "منبع" تصویر علاقه‌مند هستیم که باید لینک مجزا .jpg یا .png تصویر باشد:

In [25]:

```
1 computer['src']
```

Out[25]:

```
'//upload.wikimedia.org/wikipedia/commons/thumb/6/6f/Kasparov_Magath_1985_
Hamburg-2.png/220px-Kasparov_Magath_1985_Hamburg-2.png'
```

ما می‌توانیم آن را با یک سلول مارک‌دان با استفاده از موارد زیر نمایش دهیم:

```
<img src='https://upload.wikimedia.org/wikipedia/commons/thumb/b/be/Deep_Blue.j
pg/220px-Deep_Blue.jpg'>
```

حال که لینک واقعی src را دارید، می‌توانید با استفاده از requests تصویر را دریافت کنید و با ویژگی .content آن را دریافت کنید. توجه کنید که باید قبل از لینک https:// را اضافه کنید، اگر این کار را انجام ندهید، requests اعتراض خواهد کرد (اما یک کد خطای خیلی شرح‌دهنده را نمی‌دهد).

In [26]:

```
1 image_link = requests.get('https://upload.wikimedia.org/wikipedia/commons/thumb/b/be
```

In [27]:

```
# The raw content (its a binary file, meaning we will need to use binary read/write
image_link.content
```

```
5S\nv\x13\xc3\xc0\x0c=\xb1\x1dM=\xa7\xa53`",\x1a\xf3\x9ayM1\x03S\x15\xa
5\xd6$V\x9c:S\x00Ii\xfdm\xc3\xe5]M\x86\x96\x8d{\xa7\xed\xaeZ\xcfW\xdd5
\xd5\xd9\xe9j\xcf\xee\x0f\xba\x96dE^\xe1\xae\x10\xe2^\x0f\xc2\xc9\xe2\x
de\'s\x89/q\xdbt>\xda\x1b\xb7E\xbb6hoR\x10\x84\xe8\xac\xe0\x82U?\xa3\x1
1\xd4\xbc4g\xbc\x0f\x81\xaeC\x03\xe2\x8e$\xba\xc7\x19v\xdb\x96\xb6\x9c5
\x85\xa9\xcf\xae\xbb\xc6\xb1\xd1qvZ2U\xd9\xdb5)B\x893\x04\xc6\xf5\xd55)
\xdb\xdd\x03j\xf2\xdcyv=\x84\xf4\x16$\x89\xd3j=$h7\x03\xad\'JOS\xa7Y\xa
5-\xa6\x80\xad\x87\'Ph\xd4\x8d|\xa8-\x80\x04t\xa3\x906\xcb\xa9;P\x00\xa
7\x0f1\x7fj\xae\xbd\xb2\x0f\xfaB\xbc\xcd\xe2$z\xaf\x16\xf1+AZ7\x8b\xdf
\r\xb6\x8b\x85\xff\x00\n\xbfX\xff\x006\xb8O\x85\x91\x89)X\xbd\xa5\xf61
\x85\x95\x0f\xa2\xd9R\x8b\xae>\x0e\x8d\xed\x03X\x93:\t\xaf=\xb1k\xcb\x9
b\xbe%\xc6\x9f\xc4R\x11wu{qp\xfaP\x02R\x16\xb7\n\xcc\x0e\x83\xbd#\xca\x
af\xc3M\x94g\x8c\x92V\x83R\xe9\x0e8\x01\xd9\xb5\x14\xcfC\x14P+\x01+PPA$
%]\t\x11\xa5\x01\xb7\x07l\xb5lKj\x04\x7fd\xcc\xfd\x94Wn\xa2\x023\x12\x9
0\xa2@\'m\x04\xc7\xca\xb5\x99\x03\x83\x80\xab/xF\xbbm\xe7B\x0b\x12\'@\x
0cx\xd2\\\xe0\x88\xd2F\xb3\x9a\x87 \xe9 \x98\xd2w\xa2\x00\xd5\xac\x16\x
c8\x981\x1bS^\x04\xa1\xean\xe4)\x80\xb5~\x8ct\x14\xbdR$\x02\x8f\x9c\xd3
^\x06S\xea\x8e%$\x0f\xacTA\xf2\x15:\x04x\xed\x11\x1a\xc0&\x0e\xa0\xc5gj
\x90bD\xc6\xbaE\x15\x99pr\x10\x93:\x9c\xda\x9a\xd1Q\xd0\x82\x15\x07\xc6
c\xca\x80\x03\n\x93\x98\xea4=?\x9d(\xbc\xc0\x15\x19\x00\xec`\xd1\x8f\xd
```

بیایید این را به یک فایل بنویسیم: به‌کارگیری 'wb' برای نشان‌دهی نوشتار باینری فایل.

In [28]:

```
f = open('my_new_file_name.jpg','wb')
```

In [29]:

```
f.write(image_link.content)
```

Out[29]:

```
18448
```

In [30]:

```
f.close()
```

حال می‌توانیم این فایل را در این نوت‌بوک به صورت مارک‌دانک نمایش دهیم با استفاده از:

```
<img src="'my_new_file_name.jpg'>
```

فقط این خط را در یک سلول جدید مارک‌دانک بنویسید و تصویری که دریافت کردیم را نمایش خواهد داد!

پروژه نمونه - کار با صفحات و آیتم‌های متعدد

بیایید مثالی واقع‌گرایانه‌تر از جمع‌آوری اطلاعات یک سایت را نشان دهیم. وبسایت http://books.toscrape.com/index.html (http://books.toscrape.com/index.html) به‌طور خاص برای جمع‌آوری اطلاعات طراحی شده است. بیایید سعی کنیم عنوان هر کتابی که دارای امتیاز 2 ستاره است را دریافت کنیم و در نهایت فقط یک لیست پایتونی با عناوین آن‌ها را داشته باشیم.

ما مراحل زیر را انجام خواهیم داد:

- شناسایی ساختار URL برای مرور تمام صفحات
- جمع‌آوری اطلاعات از هر صفحه در کاتالوگ
- شناسایی تگ/کلاسی که امتیاز ستاره را نمایش می‌دهد
- فیلتر کردن بر اساس این امتیاز ستاره با استفاده از یک عبارت if
- ذخیره نتایج در یک لیست

می‌توانیم ببینیم که ساختار URL به صورت زیر است:

```
http://books.toscrape.com/catalogue/page-1.html
```

In [31]:

```
base_url = 'http://books.toscrape.com/catalogue/page-{}.html'
```

می‌توانیم سپس شماره صفحه را با استفاده از `.format()` پر کنیم

In [32]:

```
res = requests.get(base_url.format('1'))
```

حالا بیایید محصولات (کتاب‌ها) را از نتیجه درخواست get بگیریم:

In [33]:

```
soup = bs4.BeautifulSoup(res.text,"lxml")
```

In [34]:

```
soup.select(".product_pod")
```

```
 <p class="instock availability">
 <i class="icon-ok"></i>

         In stock

 </p>
 <form>
 <button class="btn btn-primary btn-block" data-loading-text="Addin
g..." type="submit">Add to basket</button>
 </form>
 </div>
 </article>,
 <article class="product_pod">
 <div class="image_container">
 <a href="sharp-objects_997/index.html"><img alt="Sharp Objects" class
="thumbnail" src="../media/cache/32/51/3251cf3a3412f53f339e42cac213409
3.jpg"/></a>
 </div>
 <p class="star-rating Four">
 <i class="icon-star"></i>
```

حالا می‌توانیم ببینیم که هر کتاب دارای کلاس product_pod است. ما می‌توانیم هر برچسبی با این کلاس را انتخاب کنیم و سپس آن را بر اساس امتیاز آن کاهش دهیم.

In [35]:

```
products = soup.select(".product_pod")
```

In [36]:

```
example = products[0]
```

In [37]:

```
type(example)
```

Out[37]:

```
bs4.element.Tag
```

In [38]:

```
example.attrs
```

Out[38]:

```
{'class': ['product_pod']}
```

حالا با بازرسی سایت می‌توانیم ببینیم که کلاسی که ما می‌خواهیم به شکل 'class='star-rating Two است، اگر بر روی آن در مرورگر خود کلیک کنید، متوجه خواهید شد که فضای خالی به عنوان یک . نمایش داده می‌شود، بنابراین این بدان معناست که ما باید برای "star-rating.Two." جستجو کنیم.

In [39]:

```
1 list(example.children)
```

Out[39]:

```
['\n',
 <div class="image_container">
 <a href="a-light-in-the-attic_1000/index.html"><img alt="A Light in the A
ttic" class="thumbnail" src="../media/cache/2c/da/2cdad67c44b002e7ead0cc35
693c0e8b.jpg"/></a>
 </div>,
 '\n',
 <p class="star-rating Three">
 <i class="icon-star"></i>
 <i class="icon-star"></i>
 <i class="icon-star"></i>
 <i class="icon-star"></i>
 <i class="icon-star"></i>
 </p>,
 '\n',
 <h3><a href="a-light-in-the-attic_1000/index.html" title="A Light in the
Attic">A Light in the ...</a></h3>,
 '\n',
 <div class="product_price">
 <p class="price_color">Â£51.77</p>
 <p class="instock availability">
 <i class="icon-ok"></i>

         In stock

 </p>
 <form>
 <button class="btn btn-primary btn-block" data-loading-text="Adding..." t
ype="submit">Add to basket</button>
 </form>
 </div>,
 '\n']
```

In [40]:

```
1 example.select('.star-rating.Three')
```

Out[40]:

```
[<p class="star-rating Three">
 <i class="icon-star"></i>
 <i class="icon-star"></i>
 <i class="icon-star"></i>
 <i class="icon-star"></i>
 <i class="icon-star"></i>
 </p>]
```

اما ما به دنبال 2 ستاره هستیم، پس به نظر می‌رسد می‌توانیم بررسی کنیم که آیا چیزی برگردانده شده است یا خیر.

In [41]:

```
example.select('.star-rating.Two')
```

Out[41]:

```
[]
```

به طور جایگزین، می‌توانیم به سرعت رشته متن را بررسی کنیم تا ببینیم آیا "star-rating Two" در آن وجود دارد یا خیر. هر دو رویکرد قابل قبول است (همچنین رویکردهای دیگری نیز وجود دارند!)

حالا بیایید ببینیم چگونه می‌توانیم عنوان را در صورت مطابقت با 2 ستاره دریافت کنیم:

In [42]:

```
example.select('a')
```

Out[42]:

```
[<a href="a-light-in-the-attic_1000/index.html"><img alt="A Light in the A
ttic" class="thumbnail" src="../media/cache/2c/da/2cdad67c44b002e7ead0cc35
693c0e8b.jpg"/></a>,
 <a href="a-light-in-the-attic_1000/index.html" title="A Light in the Atti
c">A Light in the ...</a>]
```

In [43]:

```
example.select('a')[1]
```

Out[43]:

```
<a href="a-light-in-the-attic_1000/index.html" title="A Light in the Atti
c">A Light in the ...</a>
```

In [44]:

```
example.select('a')[1]['title']
```

Out[44]:

```
'A Light in the Attic'
```

خوب، بیایید با ترکیب همه ایده‌هایی که درباره آن صحبت کرده‌ایم، آن را امتحان کنیم! (اجرای این کد حدود 20-60 ثانیه طول می‌کشد. لطفاً به این نکته توجه کنید که اگر دیوار آتشی مانع اجرای این اسکریپت شود. همچنین اگر خطای عدم پاسخی دریافت می‌کنید، شاید بهتر باشد یک گام تاخیر با استفاده از time.sleep(1) اضافه کنید.

In [45]:

```
two_star_titles = []

for n in range(1,51):

    scrape_url = base_url.format(n)
    res = requests.get(scrape_url)

    soup = bs4.BeautifulSoup(res.text,"lxml")
    books = soup.select(".product_pod")

    for book in books:
        if len(book.select('.star-rating.Two')) != 0:
            two_star_titles.append(book.select('a')[1]['title'])
```

In [46]:

```
two_star_titles
```

Out[46]:

```
['Starving Hearts (Triangular Trade Trilogy, #1)',
 'Libertarianism for Beginners',
 "It's Only the Himalayas",
 'How Music Works',
 'Maude (1883-1993):She Grew Up with the country',
 "You can't bury them all: Poems",
 'Reasons to Stay Alive',
 'Without Borders (Wanderlove #1)',
 'Soul Reader',
 'Security',
 'Saga, Volume 5 (Saga (Collected Editions) #5)',
 'Reskilling America: Learning to Labor in the Twenty-First Century',
 'Political Suicide: Missteps, Peccadilloes, Bad Calls, Backroom Hijin
x, Sordid Pasts, Rotten Breaks, and Just Plain Dumb Mistakes in the Ann
als of American Politics',
 'Obsidian (Lux #1)',
 'My Paris Kitchen: Recipes and Stories',
 'Masks and Shadows',
```

عالی! حالا ابزارهای لازم برای استخراج اطلاعات از هر وبسایتی که به شما علاقه‌مند است، را دارید! به خاطر داشته باشید که هر چه وبسایت پیچیده‌تر باشد، استخراج آن دشوارتر خواهد بود. همیشه اجازه بگیرید!