Beautifulsoupでスクレーピング

スクレイピングとは、ウェブサイトからデータを抽出し、必要な情報を抽出することです。 最近はインターネットにデータが多すぎるため、スクレイピングをうまく活用することが一番大事です。

パイソンでスクレイピングする時に欠かせないライブラリが"Beautiful Soup"です。 このライブラリを使えば、簡単にHTMLとXMLから情報を抽出できます。

最近スクレイピングライブラリはダウンロードからHTML分析まですべてしてくれることが多いが、BeautifulSoupはあくまでもHTMLとXMLを分析するライブラリです。 Beautiful Soup自体にはダウンロード機能がないので注意する必要があります。

Beautiful Soupをインストール

パイソンライブラリをインストールするときは、pipコマンドを使います。pipとは、パイソンパッケージ管理システムです。

パイソンパッケージは、Python Package Index(PyPI)で確認できる。 pipを使うとPyPiにあるパッケージをコマンドでインストールできる。

pipでBeautiful Soupをインストールする際は、次のようなコマンドを実行する。

```
$ pip3 install beautifulsoup4
```

ちなみにpip3はpipのパイソン3バージョンです。

BeautifulSoup 基本的な使い方

まずは、Beautifulsoupの基本的な使い方を確認してみましょう。 次のプログラムはBeautifulSoupを使って分析する簡単な例題です。 ウェブサイトからHTMLを持ってきて使うことがではなく、HTMLを文字列にして使います。 そして文字列の分析を完了すると結果を出力します。

In [1]:
```
# ライブラリの読み込み -- 1
from bs4 import BeautifulSoup

# 分析したいHTML -- 2
html = """
<html><body>
    <h1> スクレイピングとは? </h1>
    <p> ウェブページを分析すること</p>
    <p> 希望する部分を抽出すること</p>
</body></html>
"""

 # HTML分析する -- 3
soup = BeautifulSoup(html, 'html.parser')

 # 希望する部分を抽出する -- 4
h1 = soup.html.body.h1
p1 = soup.html.body.p
p2 = p1.next_sibling.next_sibling

# 要素の文字を出力する -- 5
print("h1 = " + h1.string)
print("p = " + p1.string)
print("p = " + p2.string)
```

```
h1 =  スクレイピングとは?
p =  ウェブページを分析すること
p =  希望する部分を抽出すること
```

プログラム(1)ではBeautifulsoupライブラリを読み込む。(2)では分析対象のHTMLを指定する。 プログラムの(3)ではBeautifulSoupインスタンスを生成。 この時、最初の媒介変数にHTMLを指定する。 そして二番目の媒介変数には分析する分析器(parser)の種類を指定する。 HTMLを分析するときは"html.parser"と指定する。

プログラムの(4)で希望する部分を抽出する。 正しく分析されたなら、HTMLの構造のように、ルート要素でピリオド(.)を使って値にアクセスすることができます。 コードでは"soup.html.body.h1"と記しましたが、HTML、body、h1 にある要素にアクセスしたものです。 プログラムの(5)ではstringプロパティにアクセスして要素の文字部分を抽出します。分析する際に、HTML内部には

タグが2つありますが、soup.html.body.pとアクセスすると前のタグを抽出します。 この時、一番目のnext_siblingでは 後部への改行または空白が抽出されます。 従って、next_siblingをもう一度使って2番目のタグを抽出します。
HTMLの構造を知っていれば、簡単に希望する要素を抽出できます。しかしルートから"html.body.."の形でHTML構造を一つずつ書いていくのは少し面倒で複雑です。
そのため、簡単に要素を見つける方法を紹介します。

Idで要素を探す方法

BeautifulSoupは、ルートから1つずつ要素を探す方法の他にも、idプロパティを指定して要素を探すfind()メソッドというメソッドを提供する。 それでは早速コードを見てみよう

In [2]:
```
from bs4 import BeautifulSoup

html = """
<html>
    <body>
        <h1 id= "title"> スクレイピングは? </h>
        <p> ウェブページを分析すること </p>
        <p id = "body"> 希望する部分を抽出すること</p>
    </body>
</html>
"""

# HTML分析する -- 1
soup = BeautifulSoup(html, 'html.parser')

# メソッドで希望する部分抽出する -- 2
title = soup.find(id = "title")
body = soup.find(id = "body")

# テキスト部分出力する
print("#title =" + title.string)
print("#body =" + body.string)
```

```
#title = スクレイピングとは?
#body = 希望する部分抽出すること
```

プログラムの(1)ではBeautifulSoupインスタンスを生成する。最初の媒介変数に分析したいHTMLを指定する。(2)ではidを指定して要素を抽出。 find()メソッドに"id=<値>"の形で媒介変数を指定して要素を検索する。

複数の要素を抽出する- find_all()メソッド

ちなみに複数のタグを一度に抽出したい場合はfind_all()メソッドを使います。 次のコードはHTML内部にある複数の"a"タグを抽出するプログラムです。"a" タグはハイパーリンクタグなので、リンク先はhrefプロパティで指定し、リンクを説明するテキストはタグ内部に入力します。 次のコードは説明文字とリンク先URLを抽出して出力する例です。

In [3]:
```
from bs4 import BeautifulSoup

html = """
<html>
    <body>
        <ul>
            <li><a href = "http://www.naver.com"> naver  </a></li>
            <li><a href = "http://www.daum.net"> daum </a></li>
        </ul>
    </body>
</html>
"""

# HTML分析する -- 1
soup = BeautifulSoup(html, 'html.parser')

# find_all() メソッドで抽出する -- 2
links = soup.find_all("a")

# リンク目録抽出する -- 3
for a in links:
    href = a.attrs['href']
    text = a.string
    print(text, ">", href)
```

```
naver  > http://www.naver.com
daum  > http://www.daum.net
```

プログラム(1)では、HTMLを指定してbeautifulSoupインスタンスを生成します。 プログラムの(2)ではfind_all()メソッドを使ってaタグを抽出します。

プログラムの(3)では抽出したすべての要素をfor構文で繰り返し処理します。 リンクのhrefプロパティはattrs['href']のようにattrsプロパティから抽出します。 また、内部の説明テックスはstringプロパティとして抽出します。

DOM要素のプロパティについて

では、再びDOM要素の属性を抽出する方法を確認してみましょう。 パイソンの対話型実際の環境であるREPLを使って動作を確認してみましょう。 REPLを実行するにはコマンドラインに"python3"と入力

In [6]:
```
pyhon3
>>> #コードを見やすく改行しました                    実際REPLは 別途に改行されません
>>> from bs4 import BeautifulSoup
... "<p><a href = 'a.href'>test</a></p>",
... "html.parser")

>>> #分析が正しくできたか確認する
>>> soup.prettify()
'<p>\n <a href = "a.html">\n test\n  </a>\n</p>'

>>> #attrsプロパティの資料型確認
>>> type(a.attrs)
<class 'dict'>

>>> #hrefプロパティがあるか確認
>>> 'href' in a.attrs
True

>>> #hrefプロパティ値確認
>>> a['href']
'a_html'
```

```
  File "<ipython-input-6-f88ca815835f>", line 5
    "html.parser")
                  ^
SyntaxError: invalid syntax
```

urlopen()とBeautifulsoupを組み合わせる

Beautiful Soupインスタンスを生成する方法を学びました。今まで見てきた例題のように、HTML文字列を指定することもできますが、open()関数またはurllib.request.urlopen()関数のリターン値を指定しても構いません。 それではurlopen()を使って"気象庁RSS"から特定内容を抽出してみます。

In [8]:
```
from bs4 import BeautifulSoup
import urllib.request as req

url = "http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp"

# urlopen()でデータ取得
res = req.urlopen(url)

# Beautifulsoupで分析する
soup = BeautifulSoup(res, "html.parser")

# 希望するデータ抽出する
title = soup.find("title").string
wf = soup.find("wf").string
print(title)
print(wf)
```

```
気象庁陸上中期予報
o (降水)15日(金)と16日(土)は全国で雨が降り、江原道は17日(日)まで続きます。
<br />o (気温)20日(水)までの日中の気温は、昨日(16~22度)より高い20~28度の分布となります。<br /> 一方、今回の予報期間には内陸を中心に25度以上上がるところが多く、少し暑くなるが、 <br />  16日(土)は、雨により日中の気温が上がらず、20~23度の分布となります。
```

とても簡単なプログラムとして気象庁RSSからXMLデータを抽出し、XMLの内容を出力する。プログラム(1)ではurlopen()でURLを開きます。 そしてプログラムの(2)でBeautiful Soupで分析します。 (3)で必要なタグを抽出し、結果を出力します。 ちなみに、今回のコードではプログラムを簡単に閲覧できるようにurlopen()を実行するときにwith構文を使いません。

CSS選択者の使用

Beautiful SoupはJavaScriptライブラリであるQueryのようにCSS選択者を指定して必要な要素を抽出する機能も提供する。 次はHTMLからh1タグとliタグを抽出して出力するコードです。

In [9]:
```
from bs4 import BeautifulSoup

# 分析対象 HTML -- 1
html = """
<html>
    <body>
        <div id = "meigen">
        <h1> ウィキブックス図書 </h1>
            <ul class = "items">
                <li> ユニティゲームエフェクト入門</li>
                <li>スイフトから始まるiPhoneアプリ開発教科書</li>
                <li> モダンウェブサイトデザインの定石</li>
            </ul>
        </div>
    </body>
</html>
"""

#HTML 分析する - 2
soup = BeautifulSoup(html, 'html.parser')


  必要な部分をcssクエリで 抽出する
 #タイトル部分抽出する -- 3
h1 = soup.select_one("div#meigen > h1").string
print("h1 = ", h1)

 #目録部分抽出する -- 4
li_list = soup.select("div#meigen > ul.items >  li")
for li in li_list:
    print("li = ", li.string)
```

```
h1 =     ウィキブックス図書
li =     ユニティゲームエフェクト入門
li =     スイフトから始まるiPhoneアプリ開発教科書
li =     モダンウェブサイトデザインの定石
```

プログラム(1)では分析対象のHTMLを指定します。(2)では、BeautifulSoupインスタンスを生成します。この時、内部的に分析処理が行われます。

プログラムの(3)ではselect_one()メソッドを使ってh1タグを抽出します。 この例ではh1タグが1つしかないのでh1だけ指定しても構いません。 実際にウェブサイトから抽出したHTMLで必要な要素を選択するには、CSS選択者をもう少し詳しく指定する必要があります。

そしてプログラムの(4)ではselect()メソッドを使って複数のh1タグを抽出します。 抽出した要素はリスト資料型なので、for構文を使って一つずつ確認します。

NAVER金融から為替レートを抽出

In [10]:
```
from bs4 import BeautifulSoup
import urllib.request as req

# HTML読み込み
url = "https://finance.naver.com/marketindex/"
res = req.urlopen(url)

# HTML分析する
soup = BeautifulSoup(res, "html.parser")

# 希望するデータ抽出する -- 1
price = soup.select_one("div.head_info > span.value").string
print("usd/krw =", price)
```

```
usd/krw = 1,221.00
```

In [ ]:
```

```