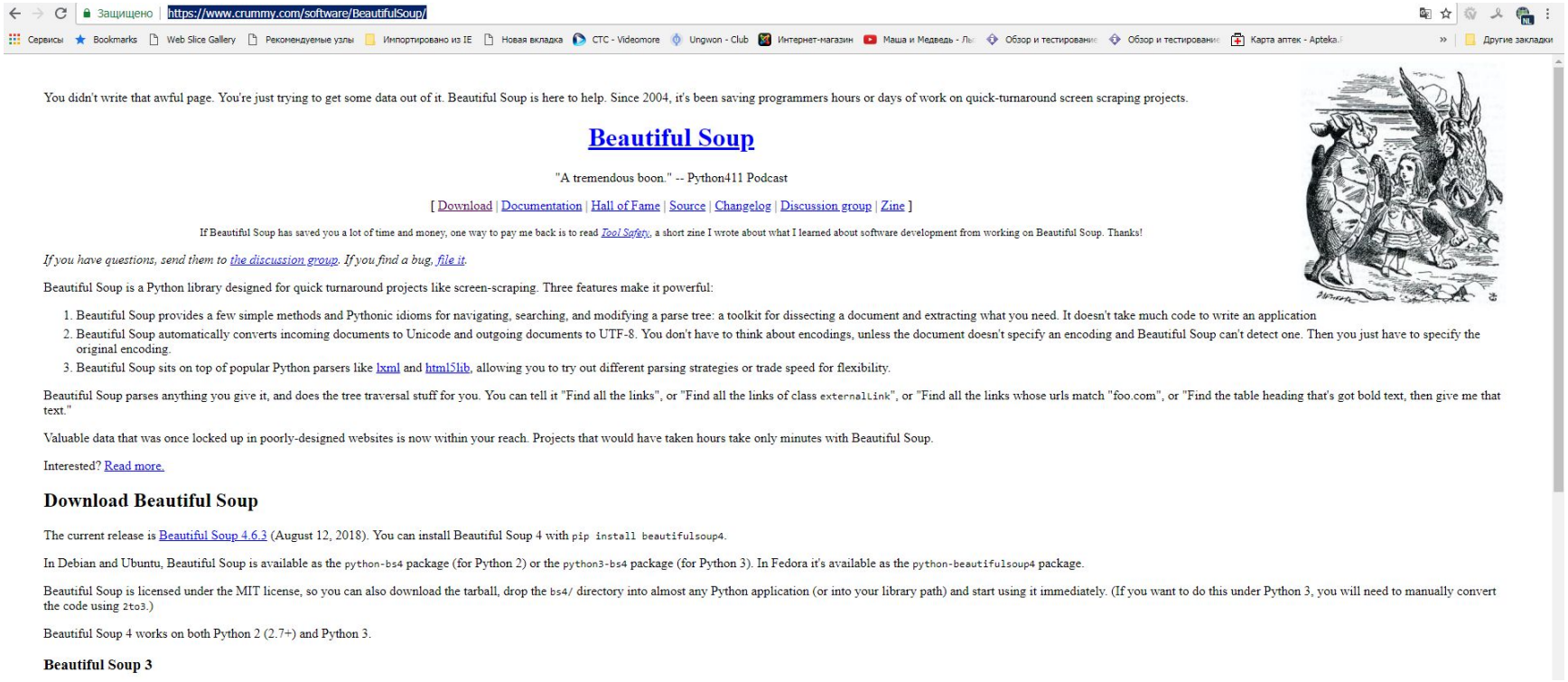
Python 01

Beautiful Soup

https://www.crummy.com/software/BeautifulSoup/



You didn't write that awful page. You're just trying to get some data out of it. Beautiful Soup is here to help. Since 2004, it's been saving programmers hours or days of work on quick-turnaround screen scraping projects.

Beautiful Soup

"A tremendous boon." -- Python411 Podcast

[Download | Documentation | Hall of Fame | Source | Changelog | Discussion group | Zine]

If Beautiful Soup has saved you a lot of time and money, one way to pay me back is to read *Tool Safety*, a short zine I wrote about what I learned about software development from working on Beautiful Soup. Thanks!

If you have questions, send them to the discussion group. If you find a bug, file it.

Beautiful Soup is a Python library designed for quick turnaround projects like screen-scraping. Three features make it powerful:

1. Beautiful Soup provides a few simple methods and Pythonic idioms for navigating, searching, and modifying a parse tree: a toolkit for dissecting a document and extracting what you need. It doesn't take much code to write an application
2. Beautiful Soup automatically converts incoming documents to Unicode and outgoing documents to UTF-8. You don't have to think about encodings, unless the document doesn't specify an encoding and Beautiful Soup can't detect one. Then you just have to specify the original encoding.
3. Beautiful Soup sits on top of popular Python parsers like lxml and html5lib, allowing you to try out different parsing strategies or trade speed for flexibility.

Beautiful Soup parses anything you give it, and does the tree traversal stuff for you. You can tell it "Find all the links", or "Find all the links of class `externalLink`", or "Find all the links whose urls match "foo.com", or "Find the table heading that's got bold text, then give me that text."

Valuable data that was once locked up in poorly-designed websites is now within your reach. Projects that would have taken hours take only minutes with Beautiful Soup.

Interested? Read more.

Download Beautiful Soup

The current release is Beautiful Soup 4.6.3 (August 12, 2018). You can install Beautiful Soup 4 with `pip install beautifulsoup4`.

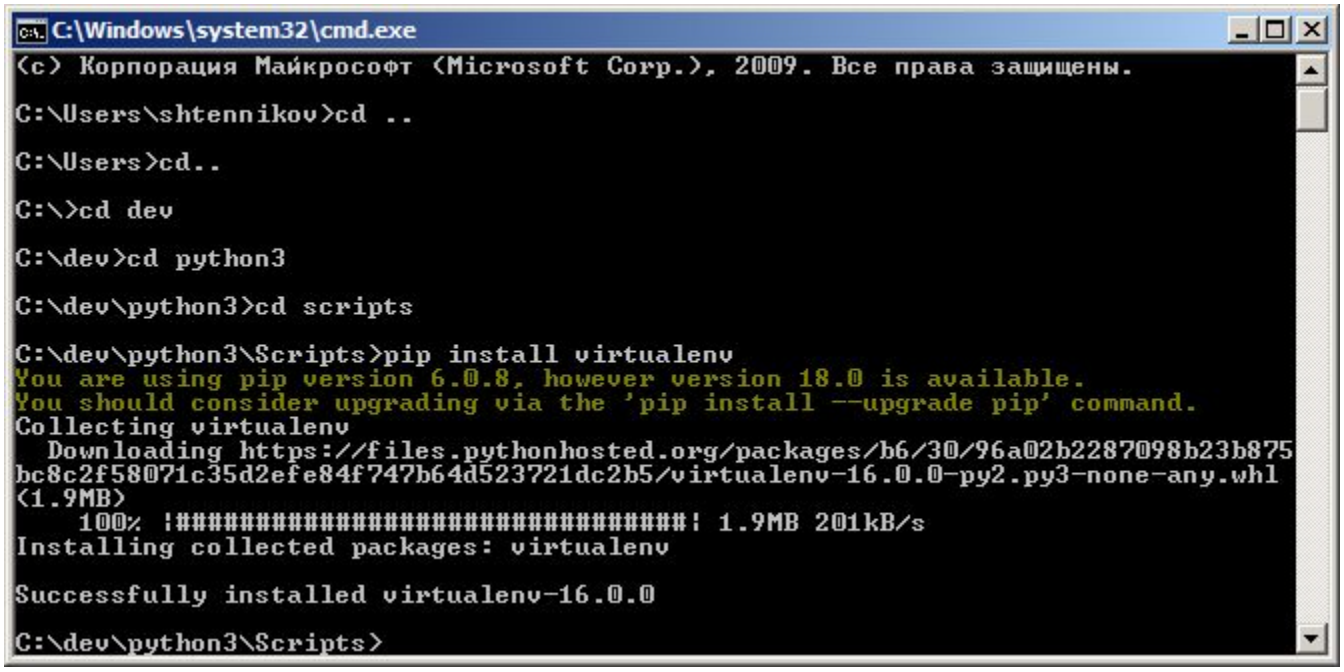
In Debian and Ubuntu, Beautiful Soup is available as the `python-bs4` package (for Python 2) or the `python3-bs4` package (for Python 3). In Fedora it's available as the `python-beautifulsoup4` package.

Beautiful Soup is licensed under the MIT license, so you can also download the tarball, drop the `bs4/` directory into almost any Python application (or into your library path) and start using it immediately. (If you want to do this under Python 3, you will need to manually convert the code using `2to3`.)

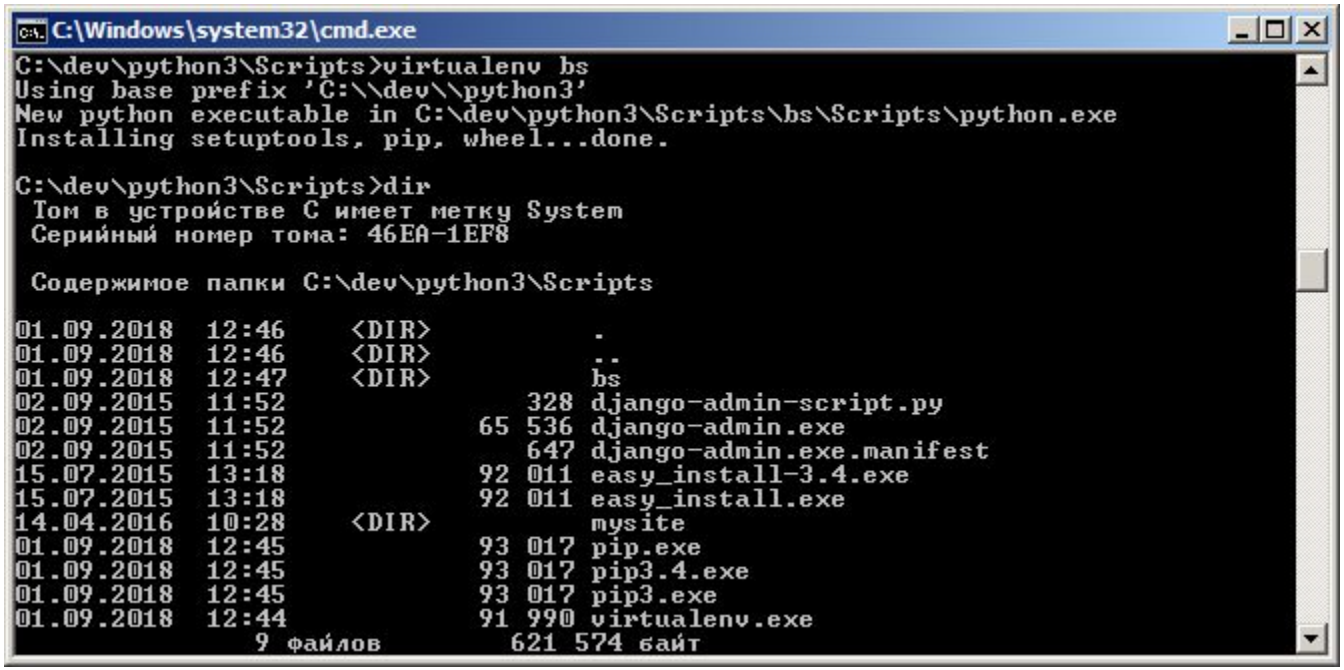
Beautiful Soup 4 works on both Python 2 (2.7+) and Python 3.

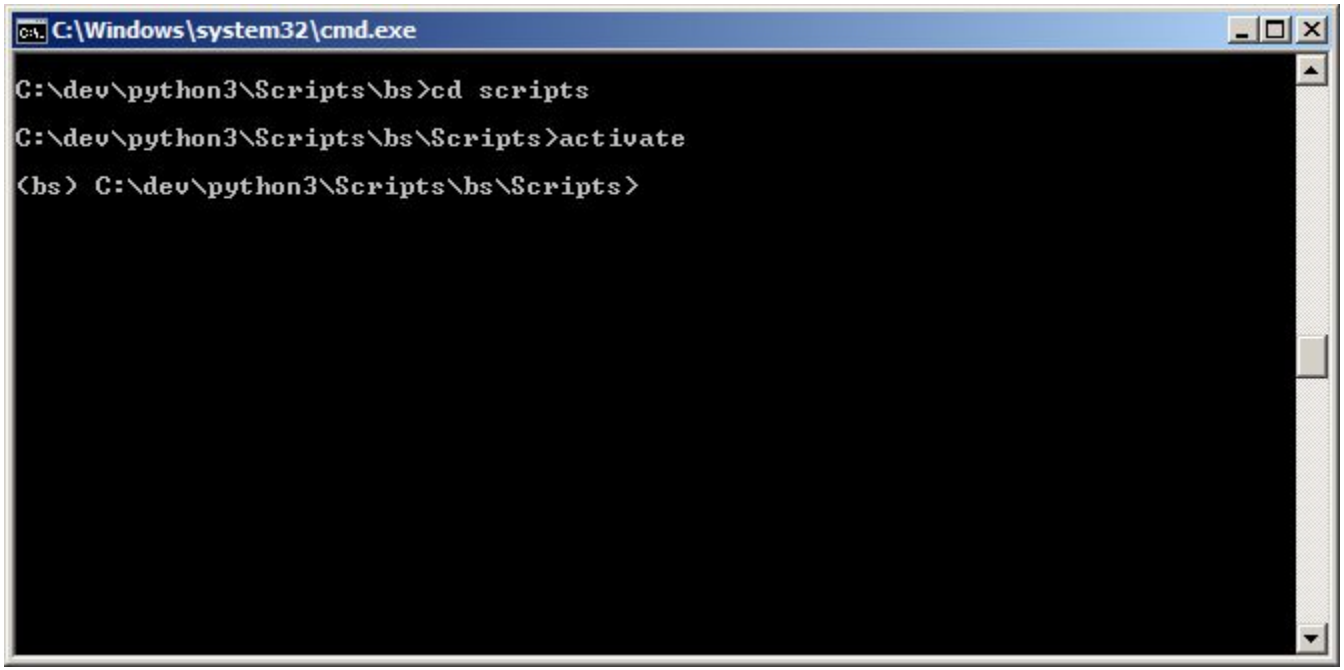
Beautiful Soup 3

virtualenv

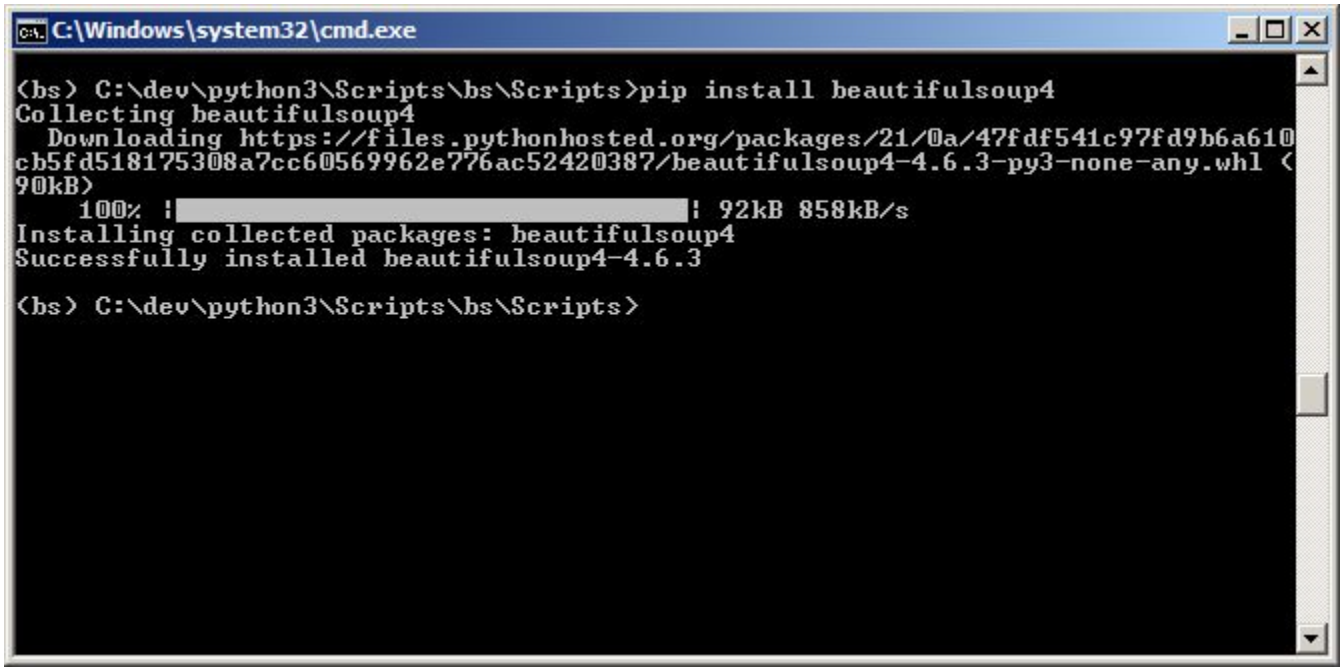

C:\Windows\system32\cmd.exe
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все права защищены.
C:\Users\shtennikov>cd ..
C:\Users>cd..
C:\>cd dev
C:\dev>cd python3
C:\dev\python3>cd scripts
C:\dev\python3\Scripts>pip install virtualenv
You are using pip version 6.0.8, however version 18.0 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
Collecting virtualenv
Downloading https://files.pythonhosted.org/packages/b6/30/96a02b2287098b23b875bc8c2f58071c35d2efe84f747b64d523721dc2b5/virtualenv-16.0.0-py2.py3-none-any.whl (1.9MB)
100% |################################| 1.9MB 201kB/s
Installing collected packages: virtualenv
Successfully installed virtualenv-16.0.0
C:\dev\python3\Scripts>

Создание виртуального окружения




C:\Windows\system32\cmd.exe
C:\dev\python3\Scripts\bs>cd scripts
C:\dev\python3\Scripts\bs\Scripts>activate
(bs) C:\dev\python3\Scripts\bs\Scripts>

pip install beautifulsoup4



pip3 install beautifulsoup4
pip3 freeze

- >>> from bs4 import BeautifulSoup as bs
- >>> import re
- >>> f_html=open('01re.html','r')
- >>> f_str=f_html.read()
- >>> #bs_str=BeautifulSoup(f_str)
- >>> bs_str=bs(f_str)
- >>> print (bs_str.prettify())

prettify()

```
>>> print (bs_str.prettify())
<!DOCTYPE html>
<html>
 <head>
  <title>
   Table
  </title>
 </head>
 <body>
  <h1>
   Title
  </h1>
  <table>
   <tr>
    <td>
     row 1 col 1
    </td>
    <td>
     row 1 col 2
    </td>
    <td>
     row 1 col 3
    </td>
   </tr>
   <tr>
    <td>
     row 2 col 1
```

Обращение к элементам

- >>> soup.contents[0].name
- 'html'
- >>> soup.contents[0].contents[0].name
- 'head'
- >>> soup.contents[0].contents[0].contents[0].name
- 'title'
- >>> soup.title
- <title>Page title</title>
- >>> soup.title.name
- 'title'
- >>> soup.title.string
- 'Page title'
- >>> soup.title.parent.name
- 'head'
- >>>

```
>>> soup.contents[0].name
'html'
>>> soup.contents[0].contents[0].name
'head'
>>> soup.contents[0].contents[0].contents[0].name
'title'
>>> soup.title
<title>Page title</title>
>>> soup.title.name
'title'
>>> soup.title.string
'Page title'
>>> soup.title.parent.name
'head'
>>>
```

find()

- >>> bs_str.find('td')
- <td>row 1 col 1</td>
- >>> bs_str.find('tr')
- <tr>
- <td>row 1 col 1</td>
- <td>row 1 col 2</td>
- <td>row 1 col 3</td>
- </tr>

- >>> soup.p
- This is paragraph one. This is paragraph two.
- >>> soup.p['align']
- 'center'
- >>> soup.a
- >>> soup.**find_all**('p')
- [This is paragraph one. This is paragraph two. , This is paragraph two.]
- >>> print(soup.**get_text()**)
- Page titleThis is paragraph one.This is paragraph two.

Чтение информации из URL

- >>> from urllib.request import urlopen
- >>> from bs4 import BeautifulSoup
- >>> html1=urlopen('https://www.djangoproject.com/download/')
- >>> html1_str=html1.read()

- html_str1

```
>>> html1_str
b'<!DOCTYPE html>\n<html lang="en">\n  <head>\n    <meta charset="utf-8">\n    <
meta http-equiv="X-UA-Compatible" content="IE=edge">\n    <meta name="viewport"
content="width=device-width, initial-scale=1">\n    <meta name="ROBOTS" content=
"ALL" />\n    <meta http-equiv="imagetoolbar" content="no" />\n    <meta name="M
SSmartTagsPreventParsing" content="true" />\n    <meta name="Copyright" content=
"Django Software Foundation" />\n    <meta name="keywords" content="Python, Djan
go, framework, open-source" />\n    <meta name="description" content="" />\n\n
  <!-- Favicons -->\n    <link rel="apple-touch-icon" href="/s/img/icon-touch.e4
```

Удобство вывода

- >>> bs1_str=BeautifulSoup(html1_str)
- >>> bs1_str

```
>>> bs1_str=BeautifulSoup(html1_str)
>>> bs1_str
<!DOCTYPE html>

<html lang="en">
<head>
<meta charset="utf-8"/>
<meta content="IE=edge" http-equiv="X-UA-Compatible"/>
<meta content="width=device-width, initial-scale=1" name="viewport"/>
<meta content="ALL" name="ROBOTS">
<meta content="no" http-equiv="imagetoolbar">
<meta content="true" name="MSSmartTagsPreventParsing">
<meta content="Django Software Foundation" name="Copyright"/>
<meta content="Python, Django, framework, open-source" name="keywords"
<meta content="" name="description"/>
<!-- Favicons -->
```

Обращение по тегам

- >>> bs1_str.h1
- <h1>Download</h1>
- >>> bs1_str.title
- <title>Download Django | Django</title>>>>> bs1_str.div
- >>> bs1_str.body
- …

```
>>> bs1_str.h1
<h1>Download</h1>
>>> bs1_str.title
<title>Download Django | Django</title>
>>>
```

Идентичность вывода

- >>> bs1_str.body.h1
- <h1>Download</h1>
- >>> bs1_str.html.body.h1
- <h1>Download</h1>
- >>> bs1_str.html.h1
- <h1>Download</h1>
- >>> bs1_str.h1
- <h1>Download</h1>

```
>>> bs1_str.body.h1
<h1>Download</h1>
>>> bs1_str.html.body.h1
<h1>Download</h1>
>>> bs1_str.html.h1
<h1>Download</h1>
>>> bs1_str.h1
<h1>Download</h1>
>>>
```

Поиск всех элементов на странице

- >>> from urllib.request import urlopen
- >>> from bs4 import BeautifulSoup as bs
- >>> import re
- >>> html1=urlopen ('https://www.djangoproject.com/download/')
- >>> htm1_str=html1.read()
- bs_str1=bs(html1_str)

```
>>> from urllib.request import urlopen
>>> from bs4 import BeautifulSoup as bs
>>> html1=urlopen('https://www.djangoproject.com/download/'
              )
>>> htm1_str=html1.read()
>>> import re
>>> bs_str1=bs(html1_str)
>>>
```

- >>> nameList=bs_str1.findAll('div')
- >>> nameList

```
>>> nameList=bs_str1.findAll('div')
>>> nameList
[<div id="top" role="banner">
<div class="container">
<a class="logo" href="https://www.djangoproject.com/">Django</a>
<p class="meta">The web framework for perfectionists with deadlines.</p>
<div role="navigation">
<ul>
```

Введение ограничения на поиск

- >>> nameList2=bs_str1.findAll('div', {'class':'footer-logo'})
- >>> nameList2
- [
- Django
-]

```
>>> nameList2=bs_str1.findAll('div', {'class':'footer-logo'})
>>> nameList2
[<div class="footer-logo">
<a class="logo" href="https://www.djangoproject.com/">Django</a>
</div>]
```

- >>> allTags=bs_str.findAll('p',{'id':'fst'})
- >>> allTags
- [ ]

```
>>> allTags=bs_str.findAll('p',{'id':'fst'})
>>> allTags
[<p id="fst"> </p>]
>>>
```

Использование регулярок для поиска

- >>> nameList3=bs_str1.findAll('div', {'class':re.compile('^footer')})
- >>> nameList3
- [
-

```
>>> nameList3=bs_str1.findAll('div', {'class':re.compile('^footer')})
>>> nameList3
[<div class="footer">
<div class="container">
<div class="footer-logo">
<a class="logo" href="https://www.djangoproject.com/">Django</a>
</div>
<ul class="thanks">
<li>
<span>Hosting by</span> <a class="rackspace" href="https://www.rackspa
ckspace</a>
</li>
<li class="design"><span>Design by</span> <a class="threespot" href="h
.threespot.com">Threespot</a> <span class="ampersand">&amp;</span> <a
revv" href="http://andrevv.com/"></a></li>
</ul>
```

Получение текста get_text()

- >>> for i in nameList4:
- print(i.**get_text()**)

```
>>> for i in nameList4:
        print(i.get_text())


Option 1: Get the latest official version
Option 2: Get the latest development version
After you get it
Supported Versions
Support Django!
For the impatient:
Which version is better?
Previous releases
Unsupported previous releases (no longer receive security updates or bug fixes)
Learn More
Get Involved
Follow Us
```

Посиск по нескольким тегам

- >>> nameList5=bs_str1.findAll(['h1','h2'])
- nameList5=bs_str1.findAll({'h1':True,'h2':True})

```
>>> nameList5=bs_str1.findAll(['h1','h2'])
>>> nameList5
[<h1>Download</h1>, <h1>How to get Django</h1>, <h2>Option 1: Get the latest
icial version</h2>, <h2>Option 2: Get the latest development version</h2>, <
fter you get it</h2>, <h2 id="supported-versions">Supported Versions</h2>, <
lass="visuallyhidden">Additional information</h1>, <h2>Support Django!</h2>,
>For the impatient:</h2>, <h2>Which version is better?</h2>, <h2>Previous re
es</h2>, <h2>Unsupported previous releases (no longer receive security updat
r bug fixes)</h2>, <h1 class="visuallyhidden">Django Links</h1>, <h2>Learn M
/h2>, <h2>Get Involved</h2>, <h2>Follow Us</h2>]
>>> nameList5=bs_str1.findAll({'h1':True,'h2':True})
>>> nameList5
[<h1>Download</h1>, <h1>How to get Django</h1>, <h2>Option 1: Get the latest
icial version</h2>, <h2>Option 2: Get the latest development version</h2>, <
fter you get it</h2>, <h2 id="supported-versions">Supported Versions</h2>, <
lass="visuallyhidden">Additional information</h1>, <h2>Support Django!</h2>,
>For the impatient:</h2>, <h2>Which version is better?</h2>, <h2>Previous re
es</h2>, <h2>Unsupported previous releases (no longer receive security updat
r bug fixes)</h2>, <h1 class="visuallyhidden">Django Links</h1>, <h2>Learn M
/h2>, <h2>Get Involved</h2>, <h2>Follow Us</h2>]
>>>
```

Вывод всего текста или список тегов

- nameList6=bs_str1.findAll(True)
- [i.name for i in nameList6]

```
>>> [i.name for i in nameList6]
['html', 'head', 'meta', 'meta', 'meta', 'meta', 'meta', 'meta', 'meta', 'meta',
 'meta', 'link', 'link', 'link', 'meta', 'meta', 'title', 'link', 'script', 'scr
ipt', 'script', 'body', 'div', 'div', 'a', 'p', 'div', 'ul', 'li', 'a', 'li', 'a
', 'li', 'a', 'li', 'a', 'li', 'a', 'li', 'a', 'li', 'a', 'li', 'a', 'div', 'div
', 'h1', 'div', 'div', 'div', 'h1', 'p', 'a', 'a', 'h2', 'p', 'a', 'a', 'pre', '
code', 'h2', 'p', 'a', 'p', 'code', 'p', 'a', 'h2', 'p', 'a', 'p', 'a', 'h2', 'p
', 'strong', 'p', 'strong', 'p', 'strong', 'p', 'a', 'img', 'hr', 'table', 'tr',
 'th', 'th', 'th', 'sup', 'a', 'th', 'sup', 'a', 'tr', 'td', 'td', 'td', 'td', '
tr', 'td', 'td', 'td', 'td', 'tr', 'td', 'sup', 'a', 'td', 'td', 'td', 'tr', 'td
', 'td', 'td', 'td', 'tr', 'td', 'td', 'td', 'td', 'tr', 'td', 'td', 'td', 'td',
 'tr', 'td', 'td', 'td', 'td', 'tr', 'td', 'td', 'td', 'td', 'tr', 'td', 'td', '
td', 'td', 'tr', 'td', 'td', 'td', 'td', 'tr', 'td', 'td', 'td', 'td', 'p', 'tab
le', 'tr', 'th', 'th', 'th', 'sup', 'a', 'th', 'sup', 'a', 'tr', 'td', 'td', 'td
', 'td', 'tr', 'td', 'td', 'td', 'td', 'tr', 'td', 'td', 'td', 'td', 'tr', 'td',
 'td', 'td', 'td', 'p', 'sup', 'br', 'sup', 'br', 'sup', 'a', 'i', 'h1', 'div',
'div', 'h2', 'div', 'img', 'div', 'ul', 'li', 'a', 'h2', 'ul', 'li', 'a', 'br',
'a', 'br', 'a', 'h2', 'p', 'a', 'a', 'p', 'h2', 'ul', 'li', 'a', 'br', 'a', 'br'
, 'a', 'li', 'a', 'br', 'a', 'br', 'a', 'h2', 'ul', 'li', 'a', 'br', 'a', 'li',
'a', 'br', 'a', 'li', 'a', 'br', 'a', 'li', 'a', 'br', 'a', 'li', 'a', 'br', 'a'
, 'li', 'a', 'br', 'a', 'li', 'a', 'br', 'a', 'li', 'a', 'br', 'a', 'li', 'a', '
br', 'a', 'li', 'a', 'br', 'a', 'li', 'a', 'br', 'a', 'div', 'div', 'div', 'h1',
 'div', 'h2', 'ul', 'li', 'a', 'li', 'a', 'li', 'a', 'li', 'a', 'li', 'a', 'li',
 'a', 'div', 'h2', 'ul', 'li', 'a', 'li', 'a', 'li', 'a', 'li', 'a', 'div', 'h2'
, 'ul', 'li', 'a', 'li', 'a', 'li', 'a', 'li', 'a', 'div', 'div', 'div', 'a', 'u
l', 'li', 'span', 'a', 'li', 'span', 'a', 'span', 'a', 'p', 'a', 'a', 'script',
'script']
>>>
```

Поиск по параметрам тега

- >>> nameList7=bs_str1.findAll(lambda tag: len(tag.**name**)==2)
- nameList7=bs_str1.findAll(lambda tag: len(tag.**get_text()**)>20)
- >>> nameList7=bs_str1.findAll(lambda tag: len(tag.**attrs**)>2)
- >>> nameList7
- [<link href="/s/img/icon-touch.e4872c4da341.png" rel="icon" sizes="192x192"/>,]

```
>>> nameList7=bs_str1.findAll(lambda tag: len(tag.attrs)>2)
>>> nameList7
[<link href="/s/img/icon-touch.e4872c4da341.png" rel="icon" sizes="192x192"/>, <
img alt="" src="/s/img/release-roadmap.e844db08610e.png" style="max-width:100%;"
/>]
```

Изменения

- >>> bs_str.body.insert(0, 'MyPage')

```
>>> bs_str.body.insert(0, 'MyPage')
>>> bs_str.body
<body>MyPage
<h1>Title</h1>
<table>
<tr>
<td>row 1 col 1</td>
<td>row 1 col 2</td>
<td>row 1 col 3</td>
</tr>
<tr>
<td>row 2 col 1</td>
<td>row 2 col 2</td>
<td>row 2 col 3</td>
</tr>
<tr>
<td>row 3 col 1</td>
<td>row 3 col 2</td>
<td>row 3 col 3</td>
</tr>
</table>
</body>
>>>
```

Доп возможности вывода

- >>> str(bs_str)
- >>> bs_str.__str__()

```
>>> str(bs_str)
'<!DOCTYPE html>\n\n<html>\n<head>\n<title>Table</title>\n</head>\n<body>MyPage\
n<h1>Title</h1>\n<table>\n<tr>\n<td>row 1 col 1</td>\n<td>row 1 col 2</td>\n<td>
row 1 col 3</td>\n</tr>\n<tr>\n<td>row 2 col 1</td>\n<td>row 2 col 2</td>\n<td>r
ow 2 col 3</td>\n</tr>\n<tr>\n<td>row 3 col 1</td>\n<td>row 3 col 2</td>\n<td>ro
w 3 col 3</td>\n</tr>\n</table>\n</body>\n</html>'
>>> bs_str.__str__()
'<!DOCTYPE html>\n\n<html>\n<head>\n<title>Table</title>\n</head>\n<body>MyPage\
n<h1>Title</h1>\n<table>\n<tr>\n<td>row 1 col 1</td>\n<td>row 1 col 2</td>\n<td>
row 1 col 3</td>\n</tr>\n<tr>\n<td>row 2 col 1</td>\n<td>row 2 col 2</td>\n<td>r
ow 2 col 3</td>\n</tr>\n<tr>\n<td>row 3 col 1</td>\n<td>row 3 col 2</td>\n<td>ro
w 3 col 3</td>\n</tr>\n</table>\n</body>\n</html>'
```

- >>> bs_str.prettify()
- >>> bs_str.renderContents()

```
>>> bs_str.prettify()
'<!DOCTYPE html>\n<html>\n <head>\n  <title>\n   Table\n  </title>\n </head>\n <
body>\n  MyPage\n  <h1>\n   Title\n  </h1>\n  <table>\n   <tr>\n    <td>\n     r
ow 1 col 1\n    </td>\n    <td>\n     row 1 col 2\n    </td>\n    <td>\n     row
 1 col 3\n    </td>\n   </tr>\n   <tr>\n    <td>\n     row 2 col 1\n    </td>\n
   <td>\n     row 2 col 2\n    </td>\n    <td>\n     row 2 col 3\n    </td>\n
</tr>\n   <tr>\n    <td>\n     row 3 col 1\n    </td>\n    <td>\n     row 3 col
2\n    </td>\n    <td>\n     row 3 col 3\n    </td>\n   </tr>\n  </table>\n </bo
dy>\n</html>'
>>> bs_str.renderContents()
b'<!DOCTYPE html>\n\n<html>\n<head>\n<title>Table</title>\n</head>\n<body>MyPage
\n<h1>Title</h1>\n<table>\n<tr>\n<td>row 1 col 1</td>\n<td>row 1 col 2</td>\n<td
>row 1 col 3</td>\n</tr>\n<tr>\n<td>row 2 col 1</td>\n<td>row 2 col 2</td>\n<td>
row 2 col 3</td>\n</tr>\n<tr>\n<td>row 3 col 1</td>\n<td>row 3 col 2</td>\n<td>r
ow 3 col 3</td>\n</tr>\n</table>\n</body>\n</html>'
>>>
```

Вывод содержимого тегов

- >>> MyTitle=bs_str.title
- >>> str(MyTitle)
- '<title>Table</title>'
- >>> MyTitle.renderContents()
- b'Table'

```
>>> MyTitle=bs_str.title
>>> str(MyTitle)
'<title>Table</title>'
>>> MyTitle.renderContents()
b'Table'
>>>
```

Присвоение имен и замена содержимого

- >>> titleTag=bs_str.html.head.title
- >>> titleTag
- <title id="Main Title">Table</title>
- >>> titleTag.string
- 'Table'
- >>> len(bs_str('title'))
- 1
- >>> titleTag['id']='Hello BS'
- >>> bs_str.html.head.title
- <title id="Hello BS">Table</title>
- >>> titleTag.contents[0]
- 'Table'
- >>> titleTag.contents[0].replaceWith('BS Table')
- 'Table'
- >>> bs_str.html.head.title
- <title id="Hello BS">BS Table</title>

```
>>> titleTag=bs_str.html.head.title
>>> titleTag
<title id="Main Title">Table</title>
>>> titleTag.string
'Table'
>>> len(bs_str('title'))
1
>>> titleTag['id']='Hello BS'
>>> bs_str.html.head.title
<title id="Hello BS">Table</title>
>>> titleTag.contents[0]
'Table'
>>> titleTag.contents[0].replaceWith('BS Table')
'Table'
>>> bs_str.html.head.title
<title id="Hello BS">BS Table</title>
>>>
```

Атрибуты тегов

- >>> fstTag,scndTag=bs_str.findAll('p')
- >>> fstTag
-
- >>> fstTag['id']
- 'fst'
- >>> scndTag['id']
- 'scnd'

```
>>> fstTag,scndTag=bs_str.findAll('p')
>>> fstTag
<p id="fst"> </p>
>>> fstTag['id']
'fst'
>>> scndTag['id']
'scnd'
>>> allTags=bs_str.findAll('p',{'id':'fst'})
>>> allTags
[<p id="fst"> </p>]
>>>
```

- >>> pTag=bs_str.p
- >>> pTag
-
- >>> pTag.contents
- [' ']
- >>> pTag.contents[0].contents
- AttributeError: 'NavigableString' object has no attribute 'contents'

- >>> bs_str1.td
- <td>2.1</td>
- >>> bs_str1.td.string
- '2.1'
- >>> bs_str1.td.contents[0]
- '2.1'
- >>> bs_str1.td.get_text()
- '2.1'

Вверх и вниз по уровню

- >>> bs_str.table.nextSibling
- '\n'
- >>> bs_str.table.previousSibling
- '\n'

Вниз и вверх

- >>> bs_str.head.next
- '\n'
- >>> bs_str.head.next.name
- >>> bs_str.head.next.next
- <title id="Hello BS">BS Table</title>
- >>> bs_str.head.next.next.name
- 'title'
- >>> bs_str.title.next.name
- >>> bs_str.title.next.next
- '\n'
- >>> bs_str.title.next.next.next
- '\n'
- >>> bs_str.head.previous.name
- >>>

Поиск следующих по уровню findPreviousSiblings findPreviousSibling

- >>> bs_str.table.tr
- >>> rTable=bs_str.table.tr
- >>> rTable.findNextSiblings('tr')

```
>>> bs_str.table.tr
<tr>
<td>row 1 col 1</td>
<td>row 1 col 2</td>
<td>row 1 col 3</td>
</tr>
>>> rTable=bs_str.table.tr
```

```
>>> rTable.findNextSiblings('tr')
[<tr>
<td>row 2 col 1</td>
<td>row 2 col 2</td>
<td>row 2 col 3</td>
</tr>, <tr>
<td>row 3 col 1</td>
<td>row 3 col 2</td>
<td>row 3 col 3</td>
</tr>]
```

- >>> rTable.findNextSiblings('tr', limit=1)

findAllNext
findNext

- findAllNext(name, attrs, text, limit, **kwargs)
- findNext(name, attrs, text, **kwargs)

findAllPrevious
findPrevious

- findAllPrevious(name, attrs, text, limit, **kwargs)
- findPrevious(name, attrs, text, **kwargs)

findParents findParent

- findParents(name, attrs, limit, **kwargs)
- findParent(name, attrs, **kwargs)

- http://wiki.python.su/%D0%94%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D0%B8%D0%B8/BeautifulSoup#A.2BBB4EQQQ9BD4EMgQ9BD4EOQ_.2BBDwENQRCBD4ENA_.2BBD8EPgQ4BEEEOgQw:_findAll.28name.2C_attrs.2C_recursive.2C_text.2C_limit.2C_.2A.2Akwargs.29
- https://www.crummy.com/software/BeautifulSoup/bs4/doc/