00A0 2203∃ 2200∀ 2286⊆ 2713x 27FA⟺ 221A$\surd$ 221B$\sqrt[3]{}$ 2295⊕ 2297⊗

beautifulsoup Documentation

4.4.0

delong

9 15, 2016

Contents

Beautiful Soup HTMLXMLPython.,,.Beautiful Soup.

BeautifulSoup4,.,,,,,.

Python2.7Python3.2

Beautiful Soup3 ,Beautiful Soup 3 ,Beautiful Soup 4, BS4

:

- .
- ()
- . ( )

CHAPTER 1

BeautifulSoup, .HTML,HTML [1]

[1] BeautifulSoupgoogle,,.

HTML. ( ):

```
html_doc = """
<html><head><title>The Dormouse's story</title></head>
<body>
<p class="title"><b>The Dormouse's story</b></p>

<p class="story">Once upon a time there were three little sisters; and␣
 ↪their names were
<a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
<a href="http://example.com/lacie" class="sister" id="link2">Lacie</a>␣
 ↪and
<a href="http://example.com/tillie" class="sister" id="link3">Tillie</
 ↪a>;
and they lived at the bottom of a well.</p>

<p class="story">...</p>
"""
```

BeautifulSoup, `BeautifulSoup` ,:

```
from bs4 import BeautifulSoup
soup = BeautifulSoup(html_doc, 'html.parser')

print(soup.prettify())
# <html>
#  <head>
#   <title>
#    The Dormouse's story
#   </title>
#  </head>
#  <body>
#   <p class="title">
```

```
#     <b>
#      The Dormouse's story
#     </b>
#    </p>
#    <p class="story">
#     Once upon a time there were three little sisters; and their names␣
 ↪were
#     <a class="sister" href="http://example.com/elsie" id="link1">
#      Elsie
#     </a>
#     ,
#     <a class="sister" href="http://example.com/lacie" id="link2">
#      Lacie
#     </a>
#     and
#     <a class="sister" href="http://example.com/tillie" id="link2">
#      Tillie
#     </a>
#     ; and they lived at the bottom of a well.
#    </p>
#    <p class="story">
#     ...
#    </p>
#  </body>
# </html>
```

:

```
soup.title
# <title>The Dormouse's story</title>

soup.title.name
# u'title'

soup.title.string
# u'The Dormouse's story'

soup.title.parent.name
# u'head'

soup.p
# <p class="title"><b>The Dormouse's story</b></p>

soup.p['class']
# u'title'

soup.a
```

```
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>

soup.find_all('a')
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]

soup.find(id="link3")
# <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>
```

<a>:

```
for link in soup.find_all('a'):
    print(link.get('href'))
    # http://example.com/elsie
    # http://example.com/lacie
    # http://example.com/tillie
```

:

```
print(soup.get_text())
# The Dormouse's story
#
# The Dormouse's story
#
# Once upon a time there were three little sisters; and their names␣
 ↪were
# Elsie,
# Lacie and
# Tillie;
# and they lived at the bottom of a well.
#
# ...
```

?,

CHAPTER 3

Beautiful Soup

Debainubuntu,:

```
$ apt-get install Python-bs4
```

Beautiful Soup 4 PyPi,, `easy_install pip . beautifulsoup4` ,Python2Python3.

```
$ easy_install beautifulsoup4
```

```
$ pip install beautifulsoup4
```

(PyPi `BeautifulSoup` ,, Beautiful Soup3 ,BS3, `BeautifulSoup` ., `beautifulsoup4`)

`easy_install pip` , BS4 ,setup.py.

```
$ Python setup.py install
```

,Beautiful SoupBS4,.

Python2.7Python3.2Beautiful Soup, Beautiful SoupPython

3.1

Beautiful SoupPython2,Python3,Python3,,.

`ImportError` : “No module named HTMLParser”, Python3Python2.

`ImportError` : “No module named html.parser”, Python2Python3.

2,BeautifulSoup4.

ROOT_TAG_NAME = u’[document]’ `SyntaxError` “Invalid syntax”,BS4PythonPython2Python3. BS4:

```
$ Python3 setup.py install
```

bs4Python

```
$ 2to3-3.2 -w bs4
```

3.2

Beautiful SoupPythonHTML,, lxml .,lxml:

```
$ apt-get install Python-lxml

$ easy_install lxml

$ pip install lxml
```

Python html5lib , html5lib,html5lib:

```
$ apt-get install Python-html5lib

$ easy_install html5lib

$ pip install html5lib
```

,:

Python	`BeautifulSoup(markup, "html.parser")`	• Python • •	• Python 2.7.3 or 3.2.2)
lxml HTML	`BeautifulSoup(markup, "lxml")`	• •	• C
lxml XML	`BeautifulSoup(markup, ["lxml-xml"])` `BeautifulSoup(markup, "xml")`	• • XML	• C
html5lib	`BeautifulSoup(markup, "html5lib")`	• • • HTML5	• •

lxml,. Python2.7.3Python33.2.2,lxmlhtml5lib, PythonHTML.

: HTMLXML,,

CHAPTER 4

BeautifulSoup ,, .

```
from bs4 import BeautifulSoup

soup = BeautifulSoup(open("index.html"))

soup = BeautifulSoup("<html>data</html>")
```

,Unicode,HTMLUnicode

```
BeautifulSoup("Sacr&eacute; bleu!")
<html><head></head><body>Sacrleu!</body></html>
```

,Beautiful Soup,Beautiful Soup.(*XML*).

CHAPTER 5

Beautiful SoupHTML,Python,4: `Tag` , `NavigableString` , `BeautifulSoup` , `Comment` .

5.1 Tag

`Tag` XMLHTMLtag:

```
soup = BeautifulSoup('<b class="boldest">Extremely bold</b>')
tag = soup.b
type(tag)
# <class 'bs4.element.Tag'>
```

Tag, .tag: nameattributes

5.1.1 Name

tag, `.name` :

```
tag.name
# u'b'
```

tagname,Beautiful SoupHTML:

```
tag.name = "blockquote"
tag
# <blockquote class="boldest">Extremely bold</blockquote>
```

5.1.2 Attributes

tag. tag `<b class="boldest">` “class” , “boldest” . tag:

```
tag['class']
# u'boldest'
```

"",:.attrs:

```
tag.attrs
# {u'class': u'boldest'}
```

tag,. , tag

```
tag['class'] = 'verybold'
tag['id'] = 1
tag
# <blockquote class="verybold" id="1">Extremely bold</blockquote>

del tag['class']
del tag['id']
tag
# <blockquote>Extremely bold</blockquote>

tag['class']
# KeyError: 'class'
print(tag.get('class'))
# None
```

HTML 4.HTML5,. class (tagCSSclass). `rel` , `rev` , `accept-charset` , `headers` , `accesskey` . Beautiful Souplist:

```
css_soup = BeautifulSoup('<p class="body strikeout"></p>')
css_soup.p['class']
# ["body", "strikeout"]

css_soup = BeautifulSoup('<p class="body"></p>')
css_soup.p['class']
# ["body"]
```

,HTML,Beautiful Soup

```
id_soup = BeautifulSoup('<p id="my id"></p>')
id_soup.p['id']
# 'my id'
```

tag,

```
rel_soup = BeautifulSoup('<p>Back to the <a rel="index">homepage</a></
↪p>')
rel_soup.a['rel']
# ['index']
rel_soup.a['rel'] = ['index', 'contents']
print(rel_soup.p)
# <p>Back to the <a rel="index contents">homepage</a></p>
```

XML,tag

```
xml_soup = BeautifulSoup('<p class="body strikeout"></p>', 'xml')
xml_soup.p['class']
# u'body strikeout'
```

5.2

tag.Beautiful Soup `NavigableString` tag:

```
tag.string
# u'Extremely bold'
type(tag.string)
# <class 'bs4.element.NavigableString'>
```

`NavigableString` PythonUnicode, `. unicode() NavigableString` Unicode:

```
unicode_string = unicode(tag.string)
unicode_string
# u'Extremely bold'
type(unicode_string)
# <type 'unicode'>
```

tag,, *replace_with()* :

```
tag.string.replace_with("No longer bold")
tag
# <blockquote>No longer bold</blockquote>
```

`NavigableString` , .,(tagtag), `.contents .string find()` .

Beautiful Soup `NavigableString` , `unicode()` ,Unicode,Beautiful Soup,..

5.3 BeautifulSoup

BeautifulSoup., Tag, .

BeautifulSoup HTMLXMLtag,nameattribute. .name, BeautifulSoup "[document]" .name

```
soup.name
# u'[document]'
```

5.4

Tag, NavigableString, BeautifulSoup htmlxml,.:

```
markup = "<b><!--Hey, buddy. Want to buy a used parser?--></b>"
soup = BeautifulSoup(markup)
comment = soup.b.string
type(comment)
# <class 'bs4.element.Comment'>
```

Comment NavigableString:

```
comment
# u'Hey, buddy. Want to buy a used parser'
```

HTML, Comment:

```
print(soup.b.prettify())
# <b>
#  <!--Hey, buddy. Want to buy a used parser?-->
# </b>
```

Beautiful SoupXML: CData, ProcessingInstruction, Declaration, Doctype. Comment, NavigableString,.CDATA:

```
from bs4 import CData
cdata = CData("A CDATA block")
comment.replace_with(cdata)

print(soup.b.prettify())
# <b>
#  <![CDATA[A CDATA block]]>
# </b>
```

””:

```
html_doc = """
<html><head><title>The Dormouse's story</title></head>
    <body>
<p class="title"><b>The Dormouse's story</b></p>

<p class="story">Once upon a time there were three little sisters; and␣
 ↪their names were
<a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
<a href="http://example.com/lacie" class="sister" id="link2">Lacie</a>␣
 ↪and
<a href="http://example.com/tillie" class="sister" id="link3">Tillie</
 ↪a>;
and they lived at the bottom of a well.</p>

<p class="story">...</p>
"""

from bs4 import BeautifulSoup
soup = BeautifulSoup(html_doc, 'html.parser')
```

6.1

TagTag,Tag.Beautiful Soup.

: Beautiful Soup,

6.1.1 tag

tagname. <head> , `soup.head` :

```
soup.head
# <head><title>The Dormouse's story</title></head>

soup.title
# <title>The Dormouse's story</title>
```

tag,tag.<body><b>:

```
soup.body.b
# <b>The Dormouse's story</b>
```

tag:

```
soup.a
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>
```

<a>,tag, *Searching the tree* ,: find_all()

```
soup.find_all('a')
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]
```

6.1.2 .contents .children

tag `.contents` tag:

```
head_tag = soup.head
head_tag
# <head><title>The Dormouse's story</title></head>

head_tag.contents
[<title>The Dormouse's story</title>]

title_tag = head_tag.contents[0]
title_tag
# <title>The Dormouse's story</title>
```

```
title_tag.contents
# [u'The Dormouse's story']
```

BeautifulSoup ,<html> BeautifulSoup :

```
len(soup.contents)
# 1
soup.contents[0].name
# u'html'
```

.contents ,:

```
text = title_tag.contents[0]
text.contents
# AttributeError: 'NavigableString' object has no attribute 'contents'
```

tag .children ,tag:

```
for child in title_tag.children:
    print(child)
    # The Dormouse's story
```

6.1.3 .descendants

.contents .children tag.,<head><title>

```
head_tag.contents
# [<title>The Dormouse's story</title>]
```

<title>: The Dormouses story, The Dormouses story<head>. .descendants tag [5] :

```
for child in head_tag.descendants:
    print(child)
    # <title>The Dormouse's story</title>
    # The Dormouse's story
```

, <head>,2:<head><head>, BeautifulSoup (<html>),:

```
len(list(soup.children))
# 1
len(list(soup.descendants))
# 25
```

[5]

6.1.4 .string

tag NavigableString ,tag .string :

```
title_tag.string
# u'The Dormouse's story'
```

tag,tag .string , .string :

```
head_tag.contents
# [<title>The Dormouse's story</title>]

head_tag.string
# u'The Dormouse's story'
```

tag,tag .string , .string None :

```
print(soup.html.string)
# None
```

6.1.5 .strings stripped_strings

tag [2] , .strings :

```
for string in soup.strings:
    print(repr(string))
    # u"The Dormouse's story"
    # u'\n\n'
    # u"The Dormouse's story"
    # u'\n\n'
    # u'Once upon a time there were three little sisters; and their␣
↪names were\n'
    # u'Elsie'
    # u',\n'
    # u'Lacie'
    # u' and\n'
    # u'Tillie'
    # u';\nand they lived at the bottom of a well.'
    # u'\n\n'
    # u'...'
    # u'\n'
```

, .stripped_strings :

[2] ,

```
for string in soup.stripped_strings:
    print(repr(string))
    # u"The Dormouse's story"
    # u"The Dormouse's story"
    # u'Once upon a time there were three little sisters; and their␣
 ↪names were'
    # u'Elsie'
    # u','
    # u'Lacie'
    # u'and'
    # u'Tillie'
    # u';\nand they lived at the bottom of a well.'
    # u'...'
```

,

6.2

,tag:tag

6.2.1 .parent

`.parent` .,<head><title>:

```
title_tag = soup.title
title_tag
# <title>The Dormouse's story</title>
title_tag.parent
# <head><title>The Dormouse's story</title></head>
```

title:<title>

```
title_tag.string.parent
# <title>The Dormouse's story</title>
```

<html> `BeautifulSoup` :

```
html_tag = soup.html
type(html_tag.parent)
# <class 'bs4.BeautifulSoup'>
```

`BeautifulSoup` `.parent` None:

```
print(soup.parent)
# None
```

6.2.2 .parents

.parents, .parents <a>.

```
link = soup.a
link
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</
↪a>
for parent in link.parents:
    if parent is None:
        print(parent)
    else:
        print(parent.name)
# p
# body
# html
# [document]
# None
```

6.3

:

```
sibling_soup = BeautifulSoup("<a><b>text1</b><c>text2</c></b></a>")
print(sibling_soup.prettify())
# <html>
#  <body>
#   <a>
#    <b>
#     text1
#    </b>
#    <c>
#     text2
#    </c>
#   </a>
#  </body>
# </html>
```

<b><c>:,<b><c>.,..

6.3.1 .next_sibling .previous_sibling

, .next_sibling .previous_sibling:

```
sibling_soup.b.next_sibling
# <c>text2</c>

sibling_soup.c.previous_sibling
# <b>text1</b>
```

<b> .next_sibling , .previous_sibling ,<b>.,<c> .previous_sibling , .next_sibling:

```
print(sibling_soup.b.previous_sibling)
# None
print(sibling_soup.c.next_sibling)
# None
```

text1text2,:

```
sibling_soup.b.string
# u'text1'

print(sibling_soup.b.string.next_sibling)
# None
```

tag .next_sibling .previous_sibling.:

```
<a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>
<a href="http://example.com/lacie" class="sister" id="link2">Lacie</a>
<a href="http://example.com/tillie" class="sister" id="link3">Tillie</
↪a>
```

<a> .next_sibling <a>,,<a><a>:

```
link = soup.a
link
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</
↪a>

link.next_sibling
# u',\n'
```

<a> .next_sibling:

```
link.next_sibling.next_sibling
# <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
↪a>
```

6.3.2 .next_siblings .previous_siblings

.next_siblings .previous_siblings:

```
for sibling in soup.a.next_siblings:
    print(repr(sibling))
    # u',\n'
    # <a class="sister" href="http://example.com/lacie" id="link2">
↪Lacie</a>
    # u' and\n'
    # <a class="sister" href="http://example.com/tillie" id="link3">
↪Tillie</a>
    # u'; and they lived at the bottom of a well.'
    # None

for sibling in soup.find(id="link3").previous_siblings:
    print(repr(sibling))
    # ' and\n'
    # <a class="sister" href="http://example.com/lacie" id="link2">
↪Lacie</a>
    # u',\n'
    # <a class="sister" href="http://example.com/elsie" id="link1">
↪Elsie</a>
    # u'Once upon a time there were three little sisters; and their␣
↪names were\n'
    # None
```

6.4

:

```
<html><head><title>The Dormouse's story</title></head>
<p class="title"><b>The Dormouse's story</b></p>
```

HTML: “<html>”,”<head>”,”<title>”,””,”<title>”,”<p>”,.Beautiful Soup.

6.4.1 .next_element .previous_element

.next_element (tag), .next_sibling ,.

<a>, .next_sibling , [2] <a>:

```
last_a_tag = soup.find("a", id="link3")
last_a_tag
# <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>

last_a_tag.next_sibling
# '; and they lived at the bottom of a well.'
```

<a> .next_element <a>,<a>,"Tillie":

```
last_a_tag.next_element
# u'Tillie'
```

,Tillie ,<a>,Tillie,,.<a>,Tillie.

.previous_element .next_element ,:

```
last_a_tag.previous_element
# u' and\n'
last_a_tag.previous_element.next_element
# <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>
```

6.4.2 .next_elements .previous_elements

.next_elements .previous_elements ,:

```
for element in last_a_tag.next_elements:
    print(repr(element))
# u'Tillie'
# u';\nand they lived at the bottom of a well.'
# u'\n\n'
# <p class="story">...</p>
# u'...'
# u'\n'
# None
```

CHAPTER 7

Beautiful Soup,2: `find()` `find_all()` .,.

:

```
html_doc = """
<html><head><title>The Dormouse's story</title></head>
<body>
<p class="title"><b>The Dormouse's story</b></p>

<p class="story">Once upon a time there were three little sisters; and␣
 ↪their names were
<a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
<a href="http://example.com/lacie" class="sister" id="link2">Lacie</a>␣
 ↪and
<a href="http://example.com/tillie" class="sister" id="link3">Tillie</
 ↪a>;
and they lived at the bottom of a well.</p>

<p class="story">...</p>
"""

from bs4 import BeautifulSoup
soup = BeautifulSoup(html_doc, 'html.parser')
```

`find_all()`

7.1

`find_all()` , [3] ,API.tagname,,.

[3] „ `filter`

7.1.1

.,Beautiful Soup,<b>:

```
soup.find_all('b')
# [<b>The Dormouse's story</b>]
```

,Beautiful SoupUTF-8,Unicode Beautiful Soup

7.1.2

,Beautiful Soup `match()` .b,<body><b>:

```
import re
for tag in soup.find_all(re.compile("^b")):
    print(tag.name)
# body
# b
```

”t”:

```
for tag in soup.find_all(re.compile("t")):
    print(tag.name)
# html
# title
```

7.1.3

,Beautiful Soup.<a><b>:

```
soup.find_all(["a", "b"])
# [<b>The Dormouse's story</b>,
#  <a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]
```

7.1.4 True

`True` ,tag,

```
for tag in soup.find_all(True):
    print(tag.name)
# html
# head
# title
# body
# p
# b
# p
# a
# a
# a
# p
```

7.1.5

„[4], True, False

, class id, True:

```
def has_class_but_no_id(tag):
    return tag.has_attr('class') and not tag.has_attr('id')
```

find_all(), <p>:

```
soup.find_all(has_class_but_no_id)
# [<p class="title"><b>The Dormouse's story</b></p>,
#  <p class="story">Once upon a time there were...</p>,
#  <p class="story">...</p>]
```

<p><a>,<a>”id”,<html><head>,<html><head>”class”.

, , . href a .

```
def not_lacie(href):
        return href and not re.compile("lacie").search(href)
soup.find_all(href=not_lacie)
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]
```

. .

[4] ,HTMLtag,

```
from bs4 import NavigableString
def surrounded_by_strings(tag):
    return (isinstance(tag.next_element, NavigableString)
            and isinstance(tag.previous_element, NavigableString))

for tag in soup.find_all(surrounded_by_strings):
    print tag.name
# p
# a
# a
# a
# p
```

7.2 find_all()

find_all(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

`find_all()` tagtag,.:

```
soup.find_all("title")
# [<title>The Dormouse's story</title>]

soup.find_all("p", "title")
# [<p class="title"><b>The Dormouse's story</b></p>]

soup.find_all("a")
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]

soup.find_all(id="link2")
# [<a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>]

import re
soup.find(string=re.compile("sisters"))
# u'Once upon a time there were three little sisters; and their names␣
 ↪were\n'
```

„ `string` `id`? `find_all("p","title")` CSS Class”title”<p>? `find_all()`

7.2.1 name

`name` `name` tag,.

:

```
soup.find_all("title")
# [<title>The Dormouse's story</title>]
```

: `name` ,,,, `True` .

7.2.2 keyword

,tag, `id` ,Beautiful Souptag"id".

```
soup.find_all(id='link2')
# [<a class="sister" href="http://example.com/lacie" id="link2">Lacie</
↪a>]
```

`href` ,Beautiful Souptag"href":

```
soup.find_all(href=re.compile("elsie"))
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
↪a>]
```

, , , *True* .

`id` tag, `id` :

```
soup.find_all(id=True)
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
↪</a>]
```

tag:

```
soup.find_all(href=re.compile("elsie"), id='link1')
# [<a class="sister" href="http://example.com/elsie" id="link1">three</
↪a>]
```

tag,HTML5 data-* :

```
data_soup = BeautifulSoup('<div data-foo="value">foo!</div>')
data_soup.find_all(data-foo="value")
# SyntaxError: keyword can't be an expression
```

find_all() attrs tag:

```
data_soup.find_all(attrs={"data-foo": "value"})
# [<div data-foo="value">foo!</div>]
```

7.2.3 CSS

CSStag,CSS class Python, class .Beautiful Soup4.1.1, class_ CSStag:

```
soup.find_all("a", class_="sister")
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]
```

class_ „„ True :

```
soup.find_all(class_=re.compile("itl"))
# [<p class="title"><b>The Dormouse's story</b></p>]

def has_six_characters(css_class):
    return css_class is not None and len(css_class) == 6

soup.find_all(class_=has_six_characters)
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]
```

tag class .CSStag,tagCSS:

```
css_soup = BeautifulSoup('<p class="body strikeout"></p>')
css_soup.find_all("p", class_="strikeout")
# [<p class="body strikeout"></p>]

css_soup.find_all("p", class_="body")
# [<p class="body strikeout"></p>]
```

`class` CSS:

```
css_soup.find_all("p", class_="body strikeout")
# [<p class="body strikeout"></p>]
```

`class` ,CSS,:

```
soup.find_all("a", attrs={"class": "sister"})
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]
```

7.2.4 string

`string` .`name` , `string` , , , *True* . :

```
soup.find_all(string="Elsie")
# [u'Elsie']

soup.find_all(string=["Tillie", "Elsie", "Lacie"])
# [u'Elsie', u'Lacie', u'Tillie']

soup.find_all(string=re.compile("Dormouse"))
[u"The Dormouse's story", u"The Dormouse's story"]

def is_the_only_string_within_a_tag(s):
    ""Return True if this string is the only child of its parent tag.""
    return (s == s.parent.string)

soup.find_all(string=is_the_only_string_within_a_tag)
# [u"The Dormouse's story", u"The Dormouse's story", u'Elsie', u'Lacie
 ↪', u'Tillie', u'...']
```

`string` ,tag.Beautiful Soup `.string` `string` tag.Elsie<a>:

```
soup.find_all("a", string="Elsie")
# [<a href="http://example.com/elsie" class="sister" id="link1">Elsie</
 ↪a>]
```

7.2.5 limit

`find_all()` ,., `limit` .SQLlimit, `limit` ,.

3tag,2,:

```
soup.find_all("a", limit=2)
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>]
```

7.2.6 recursive

tag find_all() ,Beautiful Souptag,tag, recursive=False .

:

```
<html>
 <head>
  <title>
   The Dormouse's story
  </title>
 </head>
...
```

recursive:

```
soup.html.find_all("title")
# [<title>The Dormouse's story</title>]

soup.html.find_all("title", recursive=False)
# []
```

```
<html>
        <head>
        <title>
        The Dormouse's story
    </title>
        </head>
        ...
```

<title> <html> , , <head> . Beautiful Soup <title> . recursive=False ,, <title> .

Beautiful Soup DOM. . : find_all(): name, attrs, text, limit. find_all() find() recursive.

7.3 `find_all()` tag

`find_all()` Beautiful Soup,. `BeautifulSoup` `tag` , `find_all()` ,:

```
soup.find_all("a")
soup("a")
```

:

```
soup.title.find_all(string=True)
soup.title(string=True)
```

7.4 find()

find(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

`find_all()` tag,.<body>, `find_all()` <body>, `find_all` `limit=1` `find()` .:

```
soup.find_all('title', limit=1)
# [<title>The Dormouse's story</title>]

soup.find('title')
# <title>The Dormouse's story</title>
```

`find_all()` , `find()` .

`find_all()` , `find()` , `None` .

```
print(soup.find("nosuchtag"))
# None
```

`soup.head.title` *tag* .tag `find()` :

```
soup.head.title
# <title>The Dormouse's story</title>

soup.find("head").find("title")
# <title>The Dormouse's story</title>
```

7.5 find_parents() find_parent()

find_parents(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

find_parent(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

`find_all()` `find()` ,Beautiful Soup10API. `find_all()` ,5 `find()` ..

: `find_all()` `find()` ,. `find_parents()` `find_parent()` ,tag,. :

```
a_string = soup.find(string="Lacie")
a_string
# u'Lacie'

a_string.find_parents("a")
# [<a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>]

a_string.find_parent("p")
# <p class="story">Once upon a time there were three little sisters;␣
 ↪and their names were
#  <a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a> and
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>;
#  and they lived at the bottom of a well.</p>

a_string.find_parents("p", class="title")
# []
```

<a>,.<p>,,.class”title”<p>, `find_parents()` .

`find_parent()` `find_parents()` *.parent* *.parents* .. `.parents` .

7.6 find_next_siblings() find_next_sibling()

find_next_siblings(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

find_next_sibling(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

2 *.next_siblings* tag [5] tag, `find_next_siblings()` , `find_next_sibling()` tag.

```
first_link = soup.a
first_link
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>

first_link.find_next_siblings("a")
# [<a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
```

```
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]

first_story_paragraph = soup.find("p", "story")
first_story_paragraph.find_next_sibling("p")
# <p class="story">...</p>
```

7.7 find_previous_siblings() find_previous_sibling()

find_previous_siblings(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

find_previous_sibling(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

2 *.previous_siblings* tag [5] tag, `find_previous_siblings()` , `find_previous_sibling()` :

```
last_link = soup.find("a", id="link3")
last_link
# <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>

last_link.find_previous_siblings("a")
# [<a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>]

first_story_paragraph = soup.find("p", "story")
first_story_paragraph.find_previous_sibling("p")
# <p class="title"><b>The Dormouse's story</b></p>
```

7.8 find_all_next() find_next()

find_all_next(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

find_next(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

2 *.next_elements* tag [5] tag, `find_all_next()` , `find_next()` :

```
first_link = soup.a
first_link
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>
```

```
first_link.find_all_next(string=True)
# [u'Elsie', u',\n', u'Lacie', u' and\n', u'Tillie',
#  u';\nand they lived at the bottom of a well.', u'\n\n', u'...', u'\n
 ↪']

first_link.find_next("p")
# <p class="story">...</p>
```

, Elsie,<a>.,<p>,<a>.,,.

7.9 find_all_previous() find_previous()

find_all_previous(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

find_previous(*name* , *attrs* , *recursive* , *string* , ***kwargs*)

2 *.previous_elements* [5] tag, `find_all_previous()` , `find_previous()` .

```
first_link = soup.a
first_link
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>

first_link.find_all_previous("p")
# [<p class="story">Once upon a time there were three little sisters; .
 ↪..</p>,
#  <p class="title"><b>The Dormouse's story</b></p>]

first_link.find_previous("title")
# <title>The Dormouse's story</title>
```

`find_all_previous("p")` (class="title"),,<p><a>.,<a><p>,<p><a>,<p><a>.

7.10 CSS

Beautiful SoupCSS http://www.w3.org/TR/CSS2/selector.html [6] , `Tag` `BeautifulSoup` `.select()` , CSStag:

```
soup.select("title")
# [<title>The Dormouse's story</title>]
```

[6] CSS, http://www.w3school.com.cn/css/css_selector_type.asp

```
soup.select("p nth-of-type(3)")
# [<p class="story">...</p>]
```

tag:

```
soup.select("body a")
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie"  id="link2">Lacie
 ↪</a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]

soup.select("html head title")
# [<title>The Dormouse's story</title>]
```

tag [6] :

```
soup.select("head > title")
# [<title>The Dormouse's story</title>]

soup.select("p > a")
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie"  id="link2">Lacie
 ↪</a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]

soup.select("p > a:nth-of-type(2)")
# [<a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>]

soup.select("p > #link1")
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>]

soup.select("body > a")
# []
```

:

```
soup.select("#link1 ~ .sister")
# [<a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie"  id="link3">
 ↪Tillie</a>]
```

```
soup.select("#link1 + .sister")
# [<a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>]
```

CSS:

```
soup.select(".sister")
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]

soup.select("[class~=sister]")
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]
```

tagid:

```
soup.select("#link1")
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>]

soup.select("a#link2")
# [<a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>]
```

CSS:

```
soup.select("#link1,#link2")
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>]
```

:

```
soup.select('a[href]')
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
```

```
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]
```

:

```
soup.select('a[href="http://example.com/elsie"]')
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>]

soup.select('a[href^="http://example.com/"]')
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</
 ↪a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]

soup.select('a[href$="tillie"]')
# [<a class="sister" href="http://example.com/tillie" id="link3">Tillie
 ↪</a>]

soup.select('a[href*=".com/el"]')
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>]
```

:

```
multilingual_markup = """
 <p lang="en">Hello</p>
 <p lang="en-us">Howdy, y'all</p>
 <p lang="en-gb">Pip-pip, old fruit</p>
 <p lang="fr">Bonjour mes amis</p>
"""
multilingual_soup = BeautifulSoup(multilingual_markup)
multilingual_soup.select('p[lang|=en]')
# [<p lang="en">Hello</p>,
#  <p lang="en-us">Howdy, y'all</p>,
#  <p lang="en-gb">Pip-pip, old fruit</p>]
```

```
soup.select_one(".sister")
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</
 ↪a>
```

CSS.Beautiful SoupCSSAPI, CSS, `lxml` , ,CSS,Beautiful SoupCSSAPI.

CHAPTER 8

Beautiful Soup,

8.1 tag

Attributes ,. tag,,:

```
soup = BeautifulSoup('<b class="boldest">Extremely bold</b>')
tag = soup.b

tag.name = "blockquote"
tag['class'] = 'verybold'
tag['id'] = 1
tag
# <blockquote class="verybold" id="1">Extremely bold</blockquote>

del tag['class']
del tag['id']
tag
# <blockquote>Extremely bold</blockquote>
```

8.2 .string

tag `.string` ,:

```
markup = '<a href="http://example.com/">I linked to <i>example.com</i>
↪</a>'
soup = BeautifulSoup(markup)

tag = soup.a
```

```
tag.string = "New link text."
tag
# <a href="http://example.com/">New link text.</a>
```

: tagtag, .string tag

8.3 append()

Tag.append() tag,Python .append() :

```
soup = BeautifulSoup("<a>Foo</a>")
soup.a.append("Bar")

soup
# <html><head></head><body><a>FooBar</a></body></html>
soup.a.contents
# [u'Foo', u'Bar']
```

8.4 NavigableString() .new_tag()

,Python append() NavigableString :

```
soup = BeautifulSoup("<b></b>")
tag = soup.b
tag.append("Hello")
new_string = NavigableString(" there")
tag.append(new_string)
tag
# <b>Hello there.</b>
tag.contents
# [u'Hello', u' there']
```

, NavigableString , NavigableString :

```
from bs4 import Comment
new_comment = soup.new_string("Nice to see you.", Comment)
tag.append(new_comment)
tag
# <b>Hello there<!--Nice to see you.--></b>
tag.contents
# [u'Hello', u' there', u'Nice to see you.']
```

Beautiful Soup 4.2.1

tag `BeautifulSoup.new_tag()`:

```
soup = BeautifulSoup("<b></b>")
original_tag = soup.b

new_tag = soup.new_tag("a", href="http://www.example.com")
original_tag.append(new_tag)
original_tag
# <b><a href="http://www.example.com"></a></b>

new_tag.string = "Link text."
original_tag
# <b><a href="http://www.example.com">Link text.</a></b>
```

tagname,,

8.5 insert()

`Tag.insert()` `Tag.append()`, `.contents` ,.Python `.insert()`:

```
markup = '<a href="http://example.com/">I linked to <i>example.com</i>
 ↪</a>'
soup = BeautifulSoup(markup)
tag = soup.a

tag.insert(1, "but did not endorse ")
tag
# <a href="http://example.com/">I linked to but did not endorse <i>
 ↪example.com</i></a>
tag.contents
# [u'I linked to ', u'but did not endorse', <i>example.com</i>]
```

8.6 insert_before() insert_after()

`insert_before()` tag:

```
soup = BeautifulSoup("<b>stop</b>")
tag = soup.new_tag("i")
tag.string = "Don't"
soup.b.string.insert_before(tag)
soup.b
# <b><i>Don't</i>stop</b>
```

insert_after() tag:

```
soup.b.i.insert_after(soup.new_string(" ever "))
soup.b
# <b><i>Don't</i> ever stop</b>
soup.b.contents
# [<i>Don't</i>, u' ever ', u'stop']
```

8.7 clear()

Tag.clear() tag:

```
markup = '<a href="http://example.com/">I linked to <i>example.com</i>
 ↪</a>'
soup = BeautifulSoup(markup)
tag = soup.a

tag.clear()
tag
# <a href="http://example.com/"></a>
```

8.8 extract()

PageElement.extract() tag,:

```
markup = '<a href="http://example.com/">I linked to <i>example.com</i>
 ↪</a>'
soup = BeautifulSoup(markup)
a_tag = soup.a

i_tag = soup.i.extract()

a_tag
# <a href="http://example.com/">I linked to</a>

i_tag
# <i>example.com</i>

print(i_tag.parent)
None
```

2: BeautifulSoup ,tag.tag extract :

```
my_string = i_tag.string.extract()
my_string
# u'example.com'

print(my_string.parent)
# None
i_tag
# <i></i>
```

8.9 decompose()

Tag.decompose() :

```
markup = '<a href="http://example.com/">I linked to <i>example.com</i>
 ↪</a>'
soup = BeautifulSoup(markup)
a_tag = soup.a

soup.i.decompose()

a_tag
# <a href="http://example.com/">I linked to</a>
```

8.10 replace_with()

PageElement.replace_with() ,tag:

```
markup = '<a href="http://example.com/">I linked to <i>example.com</i>
 ↪</a>'
soup = BeautifulSoup(markup)
a_tag = soup.a

new_tag = soup.new_tag("b")
new_tag.string = "example.net"
a_tag.i.replace_with(new_tag)

a_tag
# <a href="http://example.com/">I linked to <b>example.net</b></a>
```

replace_with() tag,

8.11 wrap()

PageElement.wrap() tag [8] ,:

```
soup = BeautifulSoup("<p>I wish I was bold.</p>")
soup.p.string.wrap(soup.new_tag("b"))
# <b>I wish I was bold.</b>

soup.p.wrap(soup.new_tag("div"))
# <div><p><b>I wish I was bold.</b></p></div>
```

Beautiful Soup 4.0.5

8.12 unwrap()

Tag.unwrap() wrap() .tagtag,:

```
markup = '<a href="http://example.com/">I linked to <i>example.com</i>
 ↪</a>'
soup = BeautifulSoup(markup)
a_tag = soup.a

a_tag.i.unwrap()
a_tag
# <a href="http://example.com/">I linked to example.com</a>
```

replace_with() ,unwrap() tag

[8] wrap,,tagtag.tag *wrap()* tag

CHAPTER 9

9.1

`prettify()` Beautiful SoupUnicode,XML/HTML

```
markup = '<a href="http://example.com/">I linked to <i>example.com</i>
 ↪</a>'
soup = BeautifulSoup(markup)
soup.prettify()
# '<html>\n <head>\n </head>\n <body>\n  <a href="http://example.com/">
 ↪\n...'

print(soup.prettify())
# <html>
#  <head>
#  </head>
#  <body>
#   <a href="http://example.com/">
#    I linked to
#    <i>
#     example.com
#    </i>
#   </a>
#  </body>
# </html>
```

`BeautifulSoup` tag `prettify()`:

```
print(soup.a.prettify())
# <a href="http://example.com/">
#  I linked to
#  <i>
#   example.com
```

```
#   </i>
# </a>
```

9.2

„ BeautifulSoup Tag Python unicode() str() :

```
str(soup)
# '<html><head></head><body><a href="http://example.com/">I linked to
 ↪<i>example.com</i></a></body></html>'

unicode(soup.a)
# u'<a href="http://example.com/">I linked to <i>example.com</i></a>'
```

str() UTF-8, .

encode() decode() Unicode.

9.3

Beautiful SoupHTMLUnicode,&lquot;:

```
soup = BeautifulSoup("&ldquo;Dammit!&rdquo; he said.")
unicode(soup)
# u'<html><head></head><body>\u201cDammit!\u201d he said.</body></html>
 ↪'
```

,UnicodeUTF-8.HTML:

```
str(soup)
# '<html><head></head><body>\xe2\x80\x9cDammit!\xe2\x80\x9d he said.</
 ↪body></html>'
```

9.4 get_text()

tag, get_text() ,tagtag,Unicode:

```
markup = '<a href="http://example.com/">\nI linked to <i>example.com</
 ↪i>\n</a>'
soup = BeautifulSoup(markup)
```

```
soup.get_text()
u'\nI linked to example.com\n'
soup.i.get_text()
u'example.com'
```

tag:

```
# soup.get_text("|")
u'\nI linked to |example.com|\n'
```

:

```
# soup.get_text("|", strip=True)
u'I linked to|example.com'
```

.stripped_strings ,:

```
[text for text in soup.stripped_strings]
# [u'I linked to', u'example.com']
```

CHAPTER 10

HTML, `BeautifulSoup` .Beautiful Soup..

`BeautifulSoup` ,.,Beautiful Soup,: lxml, html5lib, Python.:

- : , html, xml, html5
- : , lxml, html5lib, html.parser

,.

,Beautiful Soup. lxml XML,lxml, `beautifulsoup` lxml,

10.1

Beautiful Soup,..HTMLXML,HTML:

```
BeautifulSoup("<a><b /></a>")
# <html><head></head><body><a><b></b></a></body></html>
```

HTML,

XML(XMLlxml).,,XML,<html>:

```
BeautifulSoup("<a><b /></a>", "xml")
# <?xml version="1.0" encoding="utf-8"?>
# <a><b/></a>
```

HTML,HTML,,,.

,.,lxml,</p>:

```
BeautifulSoup("<a></p>", "lxml")
# <html><body><a></a></body></html>
```

html5lib:

```
BeautifulSoup("<a></p>", "html5lib")
# <html><head></head><body><a><p></p></a></body></html>
```

html5lib</p>,,<head>.

pyhton:

```
BeautifulSoup("<a></p>", "html.parser")
# <a></a>
```

lxml [7] ,Python</p>,html5lib<body>,lxml<html>.

<a></p>,"",html5libHTML5,""."".

, `BeautifulSoup` ,,.

[7] html5lib,

HTMLXML,ASCII UTF-8,Beautiful Soup,Unicode:

```
markup = "<h1>Sacr\xc3\xa9 bleu!</h1>"
soup = BeautifulSoup(markup)
soup.h1
# <h1>Sacrleu!</h1>
soup.h1.string
# u'Sacr\xe9 bleu!'
```

(),Beautiful Soup Unicode. `BeautifulSoup .original_encoding`:

```
soup.original_encoding
'utf-8'
```

,.,,.,. `BeautifulSoup from_encoding.`

ISO-8859-8,,Beautiful SoupISO-8859-7:

```
markup = b"<h1>\xed\xe5\xec\xf9</h1>"
soup = BeautifulSoup(markup)
soup.h1
<h1>νεμω</h1>
soup.original_encoding
'ISO-8859-7'
```

`from_encoding`:

```
soup = BeautifulSoup(markup, from_encoding="iso-8859-8")
soup.h1
<h1></h1>
soup.original_encoding
'iso8859-8'
```

Unicode, . , (), `exclude_encodings` , . : , BS, , BS.

```
soup = BeautifulSoup(markup, exclude_encodings=["ISO-8859-7"])
soup.h1
<h1></h1>
soup.original_encoding
'WINDOWS-1255'
```

Windows-1255 , , Windows-1255 ISO-8859-8 , , . (exclude_encodings 4.4.0)

(UTF-8),UnicodeUnicode,REPLACEMENT CHARACTER (U+FFFD,) [9] . Beautifu Soup,Beautiful Soup UnicodeDammit BeautifulSoup .contains_replacement_characters True .Unicode. .contains_replacement_characters False ,,.

11.1

Beautiful Soup,,UTF-8,Latin-1:

```
markup = b'''
<html>
  <head>
    <meta content="text/html; charset=ISO-Latin-1" http-equiv="Content-
 ↪type" />
  </head>
  <body>
    <p>Sacr\xe9 bleu!</p>
  </body>
</html>
'''

soup = BeautifulSoup(markup)
print(soup.prettify())
# <html>
#  <head>
#   <meta content="text/html; charset=utf-8" http-equiv="Content-type"␣
 ↪/>
#  </head>
#  <body>
#   <p>
#    Sacrleu!
#   </p>
#  </body>
# </html>
```

,<meta>UTF-8.

[9] ()Beautful Soup,,,,,

UTF-8, `prettify()`:

```
print(soup.prettify("latin-1"))
# <html>
#  <head>
#   <meta content="text/html; charset=latin-1" http-equiv="Content-type
 ↪" />
# ...
```

`BeautifulSoup encode()`,Python `encode()`:

```
soup.p.encode("latin-1")
# '<p>Sacr\xe9 bleu!</p>'

soup.p.encode("utf-8")
# '<p>Sacr\xc3\xa9 bleu!</p>'
```

,XML,UnicodeSNOWMAN:

```
markup = u"<b>\N{SNOWMAN}</b>"
snowman_soup = BeautifulSoup(markup)
tag = snowman_soup.b
```

SNOWMANUTF-8(),SNOWMAN,ISO-Latin-1ASCII,SNOWMAN&#9731:

```
print(tag.encode("utf-8"))
# <b></b>

print tag.encode("latin-1")
# <b>&#9731;</b>

print tag.encode("ascii")
# <b>&#9731;</b>
```

11.2 Unicode, Dammit! (, !)

: UnicodeDammit BS, .

Beautiful Soup,,:

```
from bs4 import UnicodeDammit
dammit = UnicodeDammit("Sacr\xc3\xa9 bleu!")
print(dammit.unicode_markup)
# Sacrleu!
dammit.original_encoding
# 'utf-8'
```

Python `chardet cchardet` . ,,, ,:

```
dammit = UnicodeDammit("Sacr\xe9 bleu!", ["latin-1", "iso-8859-1"])
print(dammit.unicode_markup)
# Sacrleu!
dammit.original_encoding
# 'latin-1'
```

2Beautiful Soup

11.2.1

Unicode,Beautiful Soup [10] HTMLXML:

```
markup = b"<p>I just \x93love\x94 Microsoft Word\x92s smart quotes</p>"

UnicodeDammit(markup, ["windows-1252"], smart_quotes_to="html").
 ↪unicode_markup
# u'<p>I just &ldquo;love&rdquo; Microsoft Word&rsquo;s smart quotes</
 ↪p>'

UnicodeDammit(markup, ["windows-1252"], smart_quotes_to="xml").unicode_
 ↪markup
# u'<p>I just &#x201C;love&#x201D; Microsoft Word&#x2019;s smart quotes
 ↪</p>'
```

ASCII:

```
UnicodeDammit(markup, ["windows-1252"], smart_quotes_to="ascii").
 ↪unicode_markup
# u'<p>I just "love" Microsoft Word\'s smart quotes</p>'
```

,Beautiful Soup.,Beautiful SoupUnicode:

```
UnicodeDammit(markup, ["windows-1252"]).unicode_markup
# u'<p>I just \u201clove\u201d Microsoft Word\u2019s smart quotes</p>'
```

11.2.2

UTF-8,Windows-1252, [10] . . `UnicodeDammit.detwingle()` UTF-8,:

[10] ,microsoftword,,.

```
snowmen = (u"\N{SNOWMAN}" * 3)
quote = (u"\N{LEFT DOUBLE QUOTATION MARK}I like snowmen!\N{RIGHT␣
↪DOUBLE QUOTATION MARK}")
doc = snowmen.encode("utf8") + quote.encode("windows_1252")
```

,snowmenUTF-8,Windows-1252,snowmen,:

```
print(doc)
# I like snowmen!

print(doc.decode("windows-1252"))
# I like snowmen!
```

UTF-8 `UnicodeDecodeError` ,Windows-1252. , `UnicodeDammit.detwingle()` UTF-8,snowmen:

```
new_doc = UnicodeDammit.detwingle(doc)
print(new_doc.decode("utf8"))
# I like snowmen!
```

`UnicodeDammit.detwingle()` UTF-8Windows-1252,.

`BeautifulSoup` `UnicodeDammit` `UnicodeDammit.detwingle()` .Windows-1252UTF-8,,: I like snowmen!.

`UnicodeDammit.detwingle()` Beautiful Soup 4.1.0

NavigableString Tag HTMLXML, Beautiful Soup. , 2   BS , , : “pizza”

```
markup = "<p>I want <b>pizza</b> and more <b>pizza</b>!</p>"
soup = BeautifulSoup(markup, 'html.parser')
first_b, second_b = soup.find_all('b')
print first_b == second_b
# True

print first_b.previous_element == second_b.previous_element
# False
```

is

```
print first_b is second_b
# False
```

CHAPTER 13

Beautiful Soup

copy.copy() Tag NavigableString

```
import copy
p_copy = copy.copy(soup.p)
print p_copy
# <p>I want <b>pizza</b> and more <b>pizza</b>!</p>
```

,

```
print soup.p == p_copy
# True

print soup.p is p_copy
# False
```

,. extract() .

```
print p_copy.parent
# None
```

CHAPTER 14

<a>,.<a>. SoupStrainer „ SoupStrainer . SoupStrainer parse_only BeautifulSoup.

14.1 SoupStrainer

SoupStrainer *name* , *attrs* , *recursive* , *string* , ***kwargs* SoupStrainer

```
from bs4 import SoupStrainer

only_a_tags = SoupStrainer("a")

only_tags_with_id_link2 = SoupStrainer(id="link2")

def is_short_string(string):
    return len(string) < 10

only_short_strings = SoupStrainer(string=is_short_string)
```

SoupStrainer:

```
html_doc = """
<html><head><title>The Dormouse's story</title></head>
    <body>
<p class="title"><b>The Dormouse's story</b></p>

<p class="story">Once upon a time there were three little sisters; and␣
 ↪their names were
<a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
<a href="http://example.com/lacie" class="sister" id="link2">Lacie</a>␣
 ↪and
<a href="http://example.com/tillie" class="sister" id="link3">Tillie</
 ↪a>;
```

```
and they lived at the bottom of a well.</p>

<p class="story">...</p>
"""

print(BeautifulSoup(html_doc, "html.parser", parse_only=only_a_tags).
 ↪prettify())
# <a class="sister" href="http://example.com/elsie" id="link1">
#  Elsie
# </a>
# <a class="sister" href="http://example.com/lacie" id="link2">
#  Lacie
# </a>
# <a class="sister" href="http://example.com/tillie" id="link3">
#  Tillie
# </a>

print(BeautifulSoup(html_doc, "html.parser", parse_only=only_tags_with_
 ↪id_link2).prettify())
# <a class="sister" href="http://example.com/lacie" id="link2">
#  Lacie
# </a>

print(BeautifulSoup(html_doc, "html.parser", parse_only=only_short_
 ↪strings).prettify())
# Elsie
# ,
# Lacie
# and
# Tillie
# ...
#
```

SoupStrainer .,:

```
soup = BeautifulSoup(html_doc)
soup.find_all(only_short_strings)
# [u'\n\n', u'\n\n', u'Elsie', u',\n', u'Lacie', u' and\n', u'Tillie',
#  u'\n\n', u'...', u'\n']
```

CHAPTER 15

15.1

Beautiful Soup, `diagnose()` (Beautiful Soup 4.2.0),Beautiful Soup,,:

```
from bs4.diagnose import diagnose
data = open("bad.html").read()
diagnose(data)

# Diagnostic running on Beautiful Soup 4.2.0
# Python version 2.7.3 (default, Aug  1 2012, 05:16:07)
# I noticed that html5lib is not installed. Installing it may help.
# Found lxml version 2.3.2.0
#
# Trying to parse your data with html.parser
# Here's what html.parser did with the document:
# ...
```

`diagnose()` ,,

15.2

.,Beautiful Soup, `HTMLParser.HTMLParseError` .,Beautiful Soup.

Beautiful Soup,Beautiful Soup,Beautiful Soup.,,. .

`HTMLParser.HTMLParseError: malformed start tag` `HTMLParser.HTMLParseError: bad end tag` .Python, *lxmlhtml5lib*

Tag,Tag. `find_all()` [] , `find()` None .Python: tag. *lxmlhtml5lib*

15.3

- `SyntaxError: Invalid syntax` (: `ROOT_TAG_NAME = u'[document]'`),Python2Python3
- `ImportError: No module named HTMLParser` Python3Python2Beautiful Soup
- `ImportError: No module named html.parser` Python2Python3Beautiful Soup
- `ImportError: No module named BeautifulSoup` BeautifulSoup3Python,BeautifulSoup4 `bs4`
- `ImportError: No module named bs4` PythonBeautifulSoup4

15.4 XML

,Beautiful SoupHTML,XML, `BeautifulSoup` “xml”:

```
soup = BeautifulSoup(markup, "xml")
```

, *lxml*

15.5

- ,.lxml,html5lib, . `BeautifulSoup`
- HTML ,3tag. <TAG></TAG> <tag></tag> .tag, *XML* .

15.6

- `UnicodeEncodeError: 'charmap' codec can't encode character u'\xfoo' in position bar` (`UnicodeEncodeError`),(Beautiful Soup),(console)Unicode, Python wiki ,,Unicode, `u.encode("utf8")` UTF-8.
- `KeyError: [attr]` `tag['attr']` ,tag. `KeyError: 'href'` `KeyError: 'class'` ., `tag.get('attr')` ,Pythonkey
- `AttributeError: 'ResultSet' object has no attribute 'foo'` `find_all()` tag, `ResultSet` , `.foo` . `find()`

- `AttributeError: 'NoneType' object has no attribute 'foo'` `find()` .foo `find()`,`None.find()` `None`.

15.7

Beautiful Soup,, lxml .

,Beautiful Soup,lxml.Beautiful Souplxmlhtml5libPython.

cchardet

,,.

CHAPTER 16

Beautiful Soup 3

Beautiful Soup 3,.Beautiful Soup 3linux:

```
$ apt-get install Python-beautifulsoup
```

PyPi `BeautifulSoup`:

```
$ easy_install BeautifulSoup
$ pip install BeautifulSoup
```

Beautiful Soup 3.2.0

Beautiful Soup 3 .

16.1 BS4

Beautiful Soup 3Beautiful Soup 4—- `BeautifulSoup`:

```
from BeautifulSoup import BeautifulSoup
```

:

```
from bs4 import BeautifulSoup
```

- `ImportError` No module named BeautifulSoup,Beautiful Soup 3,Beautiful Soup 4
- `ImportError` No module named bs4,Beautiful Soup 4,Beautiful Soup 3.

BS4BS3,BS3, PEP8 .,.

BS3BS4

16.1.1

Beautiful Soup 3Python `SGMLParser` ,Python3.Beautiful Soup 4 `html.parser` ,lxml-html5lib.

`html.parser` `SGMLParser`. BS4 BS3. lxmlhtml5lib, `html.parser`,.,.

16.1.2

- `renderContents` -> `encode_contents`
- `replaceWith` -> `replace_with`
- `replaceWithChildren` -> `unwrap`
- `findAll` -> `find_all`
- `findAllNext` -> `find_all_next`
- `findAllPrevious` -> `find_all_previous`
- `findNext` -> `find_next`
- `findNextSibling` -> `find_next_sibling`
- `findNextSiblings` -> `find_next_siblings`
- `findParent` -> `find_parent`
- `findParents` -> `find_parents`
- `findPrevious` -> `find_previous`
- `findPreviousSibling` -> `find_previous_sibling`
- `findPreviousSiblings` -> `find_previous_siblings`
- `nextSibling` -> `next_sibling`
- `previousSibling` -> `previous_sibling`

Beautiful Soup:

- `BeautifulSoup(parseOnlyThese=...)` -> `BeautifulSoup(parse_only=...)`
- `BeautifulSoup(fromEncoding=...)` -> `BeautifulSoup(from_encoding=...)`

Python3,:

- `Tag.has_key()` -> `Tag.has_attr()`

,:

- `Tag.isSelfClosing` -> `Tag.is_empty_element`

3,Python.,BS3,BS4.

- UnicodeDammit.Unicode -> UnicodeDammit.Unicode_markup‘‘
- `Tag.next` -> `Tag.next_element`
- `Tag.previous` -> `Tag.previous_element`

16.1.3

PEP8,:

- `childGenerator()` -> `children`
- `nextGenerator()` -> `next_elements`
- `nextSiblingGenerator()` -> `next_siblings`
- `previousGenerator()` -> `previous_elements`
- `previousSiblingGenerator()` -> `previous_siblings`
- `recursiveChildGenerator()` -> `descendants`
- `parentGenerator()` -> `parents`

BS4:

```
for parent in tag.parentGenerator():
    ...
```

:

```
for parent in tag.parents:
    ...
```

()

BS3 `None` .bug. `None` .

BS42, *.strings stripped_strings* . `.strings` NavigableString, `.stripped_strings` Python-string.

16.1.4 XML

BS4XML `BeautifulStoneSoup` .XML, `BeautifulSoup` xml. `BeautifulSoup` `isHTML` .

Beautiful SoupXML.XML. `selfClosingTags` .Beautiful Soup,,.

16.1.5

HTMLXMLUnicode,Beautiful Soup 3,. `BeautifulSoup` `smartQuotesTo` `convertEntities`. `smart_quotes_to`,Unicode. `HTML_ENTITIES`, `XML_ENTITIES` `XHTML_ENTITIES` ..

UnicodeHTML,UTF-8, .

16.1.6

Tag.string .AB,A.stringB.string.

`class` ,.CSStag.

`find*` *string* *name* .Beautiful Soupnametag,tag *Tag.string* text..Beautiful Souptag,text.

`BeautifulSoup` markupMassage ..

, `ICantBelieveItsBeautifulSoup` `BeautifulSOAP` ..

`prettify()` Unicode,.

CHAPTER 17

: http://www.crummy.com/software/BeautifulSoup/bs4/doc/

: delong

BeautifulSoup3