

How to use XPath in Python?

Asked 16 years, 4 months ago Modified 2 years, 1 month ago

Viewed 395k times



265

What are the libraries that support XPath? Is there a full implementation? How is the library used? Where is its website?



python

xml

dom

xpath

python-2.x



Share

Improve this question

Follow

edited Jul 10, 2020 at 14:39



funnydman


11.2k ● 4 ● 44 ● 65

asked Aug 12, 2008 at 11:28




yeruham

6 I have this sneaky suspicion that the answers to this question are a bit stale now. – [Warren P](#) Nov 26, 2012 at 20:44

4 The answer by @gringo-suave looks like a good update.
stackoverflow.com/a/13504511/1450294 – [Michael Scheper](#)
Jun 18, 2013 at 8:09 

Scrapy offers [XPath selectors](#). – [cs95](#) Apr 20, 2019 at 20:04

As @WarrenP says, most of the answers here are extremely stale old Python-2.x, really out of date. Maybe this question should be tagged [python-2.x](#) – [smci](#) Jun 8, 2020 at 10:00 

11 Answers

Sorted by:

Highest score (default)



137



[libxml2](#) has a number of advantages:

1. Compliance to the [spec](#)
2. Active development and a community participation
3. Speed. This is really a python wrapper around a C implementation.
4. Ubiquity. The libxml2 library is pervasive and thus well tested.

Downsides include:

1. Compliance to the [spec](#). It's strict. Things like default namespace handling are easier in other libraries.
2. Use of native code. This can be a pain depending on your how your application is distributed / deployed. RPMs are available that ease some of this pain.

3. Manual resource handling. Note in the sample below the calls to `freeDoc()` and `xpathFreeContext()`. This is not very Pythonic.

If you are doing simple path selection, stick with [ElementTree](#) (which is included in Python 2.5). If you need full spec compliance or raw speed and can cope with the distribution of native code, go with `libxml2`.

Sample of libxml2 XPath Use

```
import libxml2

doc = libxml2.parseFile("tst.xml")
ctxt = doc.xpathNewContext()
res = ctxt.xpathEval("//*")
if len(res) != 2:
    print "xpath query: wrong node set size"
    sys.exit(1)
if res[0].name != "doc" or res[1].name != "foo":
    print "xpath query: wrong node set value"
    sys.exit(1)
doc.freeDoc()
ctxt.xpathFreeContext()
```

Sample of ElementTree XPath Use

```
from elementtree.ElementTree import ElementTree
mydoc = ElementTree(file='tst.xml')
for e in mydoc.findall('/foo/bar'):
    print e.get('title').text
```

Share Improve this answer

Follow

edited Feb 7, 2016 at 18:16



Markus Safar

6,583 ● 5 ● 31 ● 45

answered Aug 26, 2008 at 13:06



Ryan Cox

5,053 ● 2 ● 26 ● 18

9 using python 2.7.10 on osx I had to import ElementTree as
`from xml.etree.ElementTree import ElementTree`
– Ben Page Jan 11, 2016 at 14:34

because it is a C wrapper you might find difficulty deploying it to AWS Lambda unless you compile on an EC2 instance or a Docker image of AWS Linux – CpILL Jul 12, 2018 at 13:51

Strictness is not a downside. – Dragas Jul 18, 2020 at 17:12

libxml2 isn't built-in to Python, is it? How would one do this with the built-in `xml` libraries? – Eric Dand Sep 29, 2022 at 23:18



87

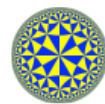
The [lxml package](#) supports xpath. It seems to work pretty well, although I've had some trouble with the self:: axis. There's also [Amara](#), but I haven't used it personally.



Share Improve this answer

Follow

edited Jan 10, 2013 at 19:26



MvG

60.7k ● 24 ● 155 ● 290

answered Aug 12, 2008 at 11:40



James Sulak

32.4k ● 11 ● 54 ● 58



1 amara's pretty nice, and one doesn't always need xpath.
– [gatoatigrado](#) Dec 5, 2009 at 5:35

3 Please add some basic details about how to use XPath with lxml. – [jpmc26](#) Oct 30, 2018 at 17:57

There's a standard library solution. I prefer fewer dependencies. See Gringo Suave's answer. – [Ted Shaneyfelt](#) Mar 30, 2021 at 5:29



82



Sounds like an lxml advertisement in here. ;)

ElementTree is included in the std library. Under 2.6 and below its xpath is pretty weak, but in 2.7+ and 3.x [much improved](#):

```
import xml.etree.ElementTree as ET

root = ET.parse(filename)
result = ''

for elem in root.findall('.//child/grandchild'):
    # How to make decisions based on attributes:
    if elem.attrib.get('name') == 'foo':
        result = elem.text
        break
```

Share Improve this answer

[edited Sep 30, 2022 at 16:28](#)

Follow

answered Nov 22, 2012 at 1:05



[Gringo Suave](#)

31.7k ● 7 ● 94 ● 81



40



Use LXML. LXML uses the full power of libxml2 and libxslt, but wraps them in more "Pythonic" bindings than the Python bindings that are native to those libraries. As such, it gets the full XPath 1.0 implementation. Native ElementTree supports a limited subset of XPath, although it may be good enough for your needs.

Share Improve this answer

answered Nov 13, 2009 at 23:11

Follow



[user210794](#)

1,011 ● 8 ● 3



33



Another option is [py-dom-xpath](#), it works seamlessly with minidom and is pure Python so works on appengine.

```
import xpath
xpath.find('//item', doc)
```

Share Improve this answer

edited Jan 27, 2013 at 14:10

Follow



[user1873471](#)

answered Jan 23, 2010 at 9:30



[Sam](#)

6,250 ● 4 ● 44 ● 53

3 Easier than lxml and libxml2 if you're already working with minidom. Works beautifully and is more "Pythonic". The `context` in the `find` function let you use another xpath result as a new search context. – [Ben](#) Oct 24, 2011 at 20:19



- 5 I too have been using py-dom-xpath as I write a plugin, because it's pure python. But I don't think it's maintained anymore, and be aware of this bug ("Cannot access an element whose name is 'text'"): code.google.com/p/py-dom-xpath/issues/detail?id=8 – Jon Coombs Feb 6, 2014 at 20:52



- 3 [py-dom-xpath seems to have been mothballed years ago in 2010](#), please at minimum edit this into your answer. – smci Jun 8, 2020 at 10:03



You can use:

15

PyXML:



```
from xml.dom.ext.reader import Sax2
from xml import xpath
doc = Sax2.FromXmlFile('foo.xml').documentElement
for url in xpath.Evaluate('//@Url', doc):
    print url.value
```



libxml2:

```
import libxml2
doc = libxml2.parseFile('foo.xml')
for url in doc.xpathEval('//@Url'):
    print url.content
```

Share Improve this answer


answered Aug 23, 2010 at 13:00

Follow

00000000
01111000
01000001
01011000

0xAX

21.8k ● 26 ● 121 ● 210

1 when I try the PyXML code, I got `ImportError: No module named ext` from `from xml.dom.ext.reader import Sax2` – [Aminah Nuraini](#) Nov 30, 2015 at 19:24 



You can use the simple `soupparser` from `lxml`

11

Example:



```
from lxml.html.soupparser import fromstring

tree = fromstring("<a>Find me!</a>")
print tree.xpath("//a/text()")
```



Share Improve this answer

edited Apr 25, 2016 at 1:59

Follow

answered Nov 15, 2015 at 5:31



[Aminah Nuraini](#)

19.1k ● 9 ● 97 ● 113

1 What difference does using soupparser make?

– [Padraic Cunningham](#) Oct 20, 2016 at 23:59

It's just an alternative – [Aminah Nuraini](#) Oct 21, 2016 at 4:43



If you want to have the power of XPATH combined with the ability to also use CSS at any point you can use

10

[parsel](#):



```
>>> from parsel import Selector
>>> sel = Selector(text=u"""<html>
    <body>
        <h1>Hello, Parsel!</h1>
        <ul>
            <li><a href="http://example.com">Link
            <li><a href="http://scrapy.org">Link 2
        </ul>
    </body>
</html>""")
>>>
>>> sel.css('h1::text').extract_first()
'Hello, Parsel!'
>>> sel.xpath('//h1/text()').extract_first()
'Hello, Parsel!'
```

Share Improve this answer

edited Apr 17, 2018 at 11:32

Follow

answered Dec 16, 2017 at 22:16



[eLRuLL](#)

18.8k ● 9 ● 77 ● 104

how should my Xpath look like if I want to get "Link 1" and "Link 2"? – [weefwefwqg3](#) Apr 17, 2018 at 5:10

- 1 for getting the text, it should be something like
`//li/a/text()` – [eLRuLL](#) Apr 17, 2018 at 11:34
-



9

The latest version of [elementtree](#) supports XPath pretty well. Not being an XPath expert I can't say for sure if the implementation is full but it has satisfied most of my needs when working in Python. I've also use lxml and



PyXML and I find etree nice because it's a standard module.



NOTE: I've since found lxml and for me it's definitely the best XML lib out there for Python. It does XPath nicely as well (though again perhaps not a full implementation).

Share Improve this answer

edited Jan 12, 2009 at 22:57

Follow

answered Aug 14, 2008 at 9:48



jkp

81.2k ● 28 ● 106 ● 104

-
- 7 ElementTree's XPath support is currently minimal at best. There are huge gaping holes in functionality, such as the lack of attribute selectors, no non-default axes, no child indexing, etc. Version 1.3 (in alpha) adds some of these features, but it's still an unashamedly partial implementation.
– James Brady Jan 9, 2009 at 23:26
-



Another library is 4Suite:

<http://sourceforge.net/projects/foursuite/>

3

I do not know how spec-compliant it is. But it has worked very well for my use. It looks abandoned.



Share Improve this answer

answered Aug 23, 2010 at 12:57



Follow



codeape

101k ● 26 ● 174 ● 200



[PyXML](#) works well.

2



You didn't say what platform you're using, however if you're on Ubuntu you can get it with `sudo apt-get install python-xml`. I'm sure other Linux distros have it as well.



If you're on a Mac, xpath is already installed but not immediately accessible. You can set `PY_USE_XMLPLUS` in your environment or do it the Python way before you import `xml.xpath`:

```
if sys.platform.startswith('darwin'):
    os.environ['PY_USE_XMLPLUS'] = '1'
```

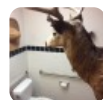
In the worst case you may have to build it yourself. This package is no longer maintained but still builds fine and works with modern 2.x Pythons. Basic docs are [here](#).

Share Improve this answer

edited Aug 12, 2008 at 20:50

Follow

answered Aug 12, 2008 at 19:34



[David Joyner](#)

23.1k ● 4 ● 30 ● 33



Highly active question. Earn 10 reputation (not counting the [association bonus](#)) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.