html5lib Documentation

Release 1.1-dev

James Graham, Geoffrey Sneddon, and contributors

Jan 20, 2018

Contents

html5lib is a pure-python library for parsing HTML. It is designed to conform to the WHATWG HTML specification, as is implemented by all major web browsers.

CHAPTER 1

Usage

Simple usage follows this pattern:

```
import html5lib
with open("mydocument.html", "rb") as f:
    document = html5lib.parse(f)
```

or:

```
import html5lib
document = html5lib.parse("<p>Hello World!")
```

By default, the `document` will be an `xml.etree` element instance. Whenever possible, html5lib chooses the accelerated `ElementTree` implementation (i.e. `xml.etree.cElementTree` on Python 2.x).

Two other tree types are supported: `xml.dom.minidom` and `lxml.etree`. To use an alternative format, specify the name of a treebuilder:

```
import html5lib
with open("mydocument.html", "rb") as f:
    lxml_etree_document = html5lib.parse(f, treebuilder="lxml")
```

When using with `urllib2` (Python 2), the charset from HTTP should be pass into html5lib as follows:

```
from contextlib import closing
from urllib2 import urlopen
import html5lib

with closing(urlopen("http://example.com/")) as f:
    document = html5lib.parse(f, transport_encoding=f.info().getparam("charset"))
```

When using with `urllib.request` (Python 3), the charset from HTTP should be pass into html5lib as follows:

```
from urllib.request import urlopen
import html5lib

```

```
with urlopen("http://example.com/") as f:
    document = html5lib.parse(f, transport_encoding=f.info().get_content_charset())
```

To have more control over the parser, create a parser object explicitly. For instance, to make the parser raise exceptions on parse errors, use:

```
import html5lib
with open("mydocument.html", "rb") as f:
    parser = html5lib.HTMLParser(strict=True)
    document = parser.parse(f)
```

When you're instantiating parser objects explicitly, pass a treebuilder class as the `tree` keyword argument to use an alternative document format:

```
import html5lib
parser = html5lib.HTMLParser(tree=html5lib.getTreeBuilder("dom"))
minidom_document = parser.parse("<p>Hello World!")
```

More documentation is available at https://html5lib.readthedocs.io/.

CHAPTER 2

Installation

html5lib works on CPython 2.7+, CPython 3.3+ and PyPy. To install it, use:

```
$ pip install html5lib
```

CHAPTER 3

Optional Dependencies

The following third-party libraries may be used for additional functionality:

- `datrie` can be used under CPython to improve parsing performance (though in almost all cases the improvement is marginal);
- `lxml` is supported as a tree format (for both building and walking) under CPython (but *not* PyPy where it is known to cause segfaults);
- `genshi` has a treewalker (but not builder); and
- `chardet` can be used as a fallback when character encoding cannot be determined.

CHAPTER 4

Bugs

Please report any bugs on the issue tracker.

CHAPTER 5

Tests

Unit tests require the `pytest` and `mock` libraries and can be run using the `py.test` command in the root directory.

Test data are contained in a separate html5lib-tests repository and included as a submodule, thus for git checkouts they must be initialized:

```
$ git submodule init
$ git submodule update
```

If you have all compatible Python implementations available on your system, you can run tests on all of them using the `tox` utility, which can be found on PyPI.

CHAPTER 6

Questions?

There's a mailing list available for support on Google Groups, html5lib-discuss, though you may get a quicker response asking on IRC in #whatwg on irc.freenode.net.

6.1 The moving parts

html5lib consists of a number of components, which are responsible for handling its features.

Parsing uses a *tree builder* to generate a *tree*, the in-memory representation of the document. Several tree representations are supported, as are translations to other formats via *tree adapters*. The tree may be translated to a token stream with a *tree walker*, from which `HTMLSerializer` produces a stream of bytes. The token stream may also be transformed by use of *filters* to accomplish tasks like sanitization.

6.1.1 Tree builders

The parser reads HTML by tokenizing the content and building a tree that the user can later access. html5lib can build three types of trees:

- `etree` - this is the default; builds a tree based on `xml.etree`, which can be found in the standard library. Whenever possible, the accelerated `ElementTree` implementation (i.e. `xml.etree.cElementTree` on Python 2.x) is used.
- `dom` - builds a tree based on `xml.dom.minidom`.
- `lxml` - uses the `lxml.etree` implementation of the `ElementTree` API. The performance gains are relatively small compared to using the accelerated `ElementTree` module.

You can specify the builder by name when using the shorthand API:

```
import html5lib
with open("mydocument.html", "rb") as f:
    lxml_etree_document = html5lib.parse(f, treebuilder="lxml")
```

To get a builder class by name, use the *getTreeBuilder()* function.

When instantiating a *HTMLParser* object, you must pass a tree builder class via the `tree` keyword attribute:

```
import html5lib
TreeBuilder = html5lib.getTreeBuilder("dom")
parser = html5lib.HTMLParser(tree=TreeBuilder)
minidom_document = parser.parse("<p>Hello World!")
```

The implementation of builders can be found in html5lib/treebuilders/.

6.1.2 Tree walkers

In addition to manipulating a tree directly, you can use a tree walker to generate a streaming view of it. html5lib provides walkers for `etree`, `dom`, and `lxml` trees, as well as `genshi` markup streams.

The implementation of walkers can be found in html5lib/treewalkers/.

html5lib provides `HTMLSerializer` for generating a stream of bytes from a token stream, and several filters which manipulate the stream.

HTMLSerializer

The serializer lets you write HTML back as a stream of bytes.

```
>>> import html5lib
>>> element = html5lib.parse('<p xml:lang="pl">Witam wszystkich')
>>> walker = html5lib.getTreeWalker("etree")
>>> stream = walker(element)
>>> s = html5lib.serializer.HTMLSerializer()
>>> output = s.serialize(stream)
>>> for item in output:
...   print("%r" % item)
'<p'
' '
'xml:lang'
'='
'pl'
'>'
'Witam wszystkich'
```

You can customize the serializer behaviour in a variety of ways. Consult the `HTMLSerializer` documentation.

Filters

html5lib provides several filters:

- *alphabeticalattributes.Filter* sorts attributes on tags to be in alphabetical order
- *inject_meta_charset.Filter* sets a user-specified encoding in the correct `<meta>` tag in the `<head>` section of the document
- *lint.Filter* raises `AssertionError` exceptions on invalid tag and attribute names, invalid PCDATA, etc.
- *optionaltags.Filter* removes tags from the token stream which are not necessary to produce valid HTML

- *sanitizer.Filter* removes unsafe markup and CSS. Elements that are known to be safe are passed through and the rest is converted to visible text. The default configuration of the sanitizer follows the WHATWG Sanitization Rules.
- *whitespace.Filter* collapses all whitespace characters to single spaces unless they're in `<pre/>` or `<textarea/>` tags.

To use a filter, simply wrap it around a token stream:

```
>>> import html5lib
>>> from html5lib.filters import sanitizer
>>> dom = html5lib.parse("<p><script>alert('Boo!')", treebuilder="dom")
>>> walker = html5lib.getTreeWalker("dom")
>>> stream = walker(dom)
>>> clean_stream = sanitizer.Filter(stream)
```

6.1.3 Tree adapters

Tree adapters can be used to translate between tree formats. Two adapters are provided by html5lib:

- *html5lib.treeadapters.genshi.to_genshi()* generates a Genshi markup stream.
- *html5lib.treeadapters.sax.to_sax()* calls a SAX handler based on the tree.

6.1.4 Encoding discovery

Parsed trees are always Unicode. However a large variety of input encodings are supported. The encoding of the document is determined in the following way:

- The encoding may be explicitly specified by passing the name of the encoding as the encoding parameter to the *parse()* method on *HTMLParser* objects.
- If no encoding is specified, the parser will attempt to detect the encoding from a `<meta>` element in the first 512 bytes of the document (this is only a partial implementation of the current HTML specification).
- If no encoding can be found and the `chardet` library is available, an attempt will be made to sniff the encoding from the byte pattern.
- If all else fails, the default encoding will be used. This is usually Windows-1252, which is a common fallback used by Web browsers.

6.2 html5lib

6.2.1 html5lib Package

HTML parsing library based on the WHATWG HTML specification. The parser is designed to be compatible with existing HTML found in the wild and implements well-defined error recovery that is largely compatible with modern desktop web browsers.

Example usage:

```
import html5lib
with open("my_document.html", "rb") as f:
    tree = html5lib.parse(f)
```

For convenience, this module re-exports the following names:

- *parse()*
- *parseFragment()*
- *HTMLParser*
- *getTreeBuilder()*
- *getTreeWalker()*
- *serialize()*

constants Module

exception html5lib.constants.**DataLossWarning**

Bases: UserWarning

Raised when the current tree is unable to represent the input data

html5parser Module

class html5lib.html5parser.**HTMLParser**(*tree=None*, *strict=False*, *namespaceHTMLElements=True*, *debug=False*)

Bases: object

HTML parser

Generates a tree structure from a stream of (possibly malformed) HTML.

__init__(*tree=None*, *strict=False*, *namespaceHTMLElements=True*, *debug=False*)

Parameters

- **tree** – a treebuilder class controlling the type of tree that will be returned. Built in treebuilders can be accessed through html5lib.treebuilders.getTreeBuilder(treeType)
- **strict** – raise an exception when a parse error is encountered
- **namespaceHTMLElements** – whether or not to namespace HTML elements
- **debug** – whether or not to enable debug mode which logs things

Example:

```
>>> from html5lib.html5parser import HTMLParser
>>> parser = HTMLParser()                     # generates parser with etree␣
↪builder
>>> parser = HTMLParser('lxml', strict=True)  # generates parser with lxml␣
↪builder which is strict
```

documentEncoding

Name of the character encoding that was used to decode the input stream, or None if that is not determined yet

parse(*stream*, **args*, ***kwargs*)

Parse a HTML document into a well-formed tree

Parameters

- **stream** – a file-like object or string containing the HTML to be parsed

  The optional encoding parameter must be a string that indicates the encoding. If specified, that encoding will be used, regardless of any BOM or later declaration (such as in a meta element).

- **scripting** – treat noscript elements as if JavaScript was turned on

Returns parsed tree

Example:

```
>>> from html5lib.html5parser import HTMLParser
>>> parser = HTMLParser()
>>> parser.parse('<html><body><p>This is a doc</p></body></html>')
<Element u'{http://www.w3.org/1999/xhtml}html' at 0x7feac4909db0>
```

parseFragment (*stream*, **args*, ***kwargs*)

Parse a HTML fragment into a well-formed tree fragment

Parameters

- **container** – name of the element we're setting the innerHTML property if set to None, default to 'div'
- **stream** – a file-like object or string containing the HTML to be parsed

  The optional encoding parameter must be a string that indicates the encoding. If specified, that encoding will be used, regardless of any BOM or later declaration (such as in a meta element)

- **scripting** – treat noscript elements as if JavaScript was turned on

Returns parsed tree

Example:

```
>>> from html5lib.html5libparser import HTMLParser
>>> parser = HTMLParser()
>>> parser.parseFragment('<b>this is a fragment</b>')
<Element u'DOCUMENT_FRAGMENT' at 0x7feac484b090>
```

exception html5lib.html5parser.**ParseError**

Bases: Exception

Error in parsed document

html5lib.html5parser.**parse** (*doc*, *treebuilder='etree'*, *namespaceHTMLElements=True*, ***kwargs*)

Parse an HTML document as a string or file-like object into a tree

Parameters

- **doc** – the document to parse as a string or file-like object
- **treebuilder** – the treebuilder to use when parsing
- **namespaceHTMLElements** – whether or not to namespace HTML elements

Returns parsed tree

Example:

```
>>> from html5lib.html5parser import parse
>>> parse('<html><body><p>This is a doc</p></body></html>')
<Element u'{http://www.w3.org/1999/xhtml}html' at 0x7feac4909db0>
```

html5lib.html5parser.**parseFragment**(*doc*, *container='div'*, *treebuilder='etree'*, *namespaceHTMLElements=True*, ***kwargs*)

Parse an HTML fragment as a string or file-like object into a tree

Parameters

- **doc** – the fragment to parse as a string or file-like object
- **container** – the container context to parse the fragment in
- **treebuilder** – the treebuilder to use when parsing
- **namespaceHTMLElements** – whether or not to namespace HTML elements

Returns parsed tree

Example:

```
>>> from html5lib.html5libparser import parseFragment
>>> parseFragment('<b>this is a fragment</b>')
<Element u'DOCUMENT_FRAGMENT' at 0x7feac484b090>
```

serializer Module

exception html5lib.serializer.**SerializeError**

Bases: Exception

Error in serialized tree

html5lib.serializer.**serialize**(*input*, *tree='etree'*, *encoding=None*, ***serializer_opts*)

Serializes the input token stream using the specified treewalker

Parameters

- **input** – the token stream to serialize
- **tree** – the treewalker to use
- **encoding** – the encoding to use
- **serializer_opts** – any options to pass to the html5lib.serializer.HTMLSerializer that gets created

Returns the tree serialized as a string

Example:

```
>>> from html5lib.html5parser import parse
>>> from html5lib.serializer import serialize
>>> token_stream = parse('<html><body><p>Hi!</p></body></html>')
>>> serialize(token_stream, omit_optional_tags=False)
'<html><head></head><body><p>Hi!</p></body></html>'
```

html5lib.serializer.**xmlcharrefreplace_errors**()

Implements the 'xmlcharrefreplace' error handling, which replaces an unencodable character with the appropriate XML character reference.

Subpackages

filters Package

`base` Module

`alphabeticalattributes` Module

class `html5lib.filters.alphabeticalattributes.`**`Filter`**(*source*)

Bases: `html5lib.filters.base.Filter`

Alphabetizes attributes for elements

`inject_meta_charset` Module

class `html5lib.filters.inject_meta_charset.`**`Filter`**(*source*, *encoding*)

Bases: `html5lib.filters.base.Filter`

Injects `<meta charset=ENCODING>` tag into head of document

`__init__`(*source*, *encoding*)

Creates a Filter

Parameters

- **`source`** – the source token stream
- **`encoding`** – the encoding to set

`lint` Module

class `html5lib.filters.lint.`**`Filter`**(*source*, *require_matching_tags=True*)

Bases: `html5lib.filters.base.Filter`

Lints the token stream for errors

If it finds any errors, it'll raise an `AssertionError`.

`__init__`(*source*, *require_matching_tags=True*)

Creates a Filter

Parameters

- **`source`** – the source token stream
- **`require_matching_tags`** – whether or not to require matching tags

`optionaltags` Module

class `html5lib.filters.optionaltags.`**`Filter`**(*source*)

Bases: `html5lib.filters.base.Filter`

Removes optional tags from the token stream

`sanitizer` Module

class `html5lib.filters.sanitizer.`**`Filter`**(*source*, *allowed_elements=frozenset({('http://www.w3.org/1999/xhtml', 'h5'), ('http://www.w3.org/2000/svg', 'metadata'), ('http://www.w3.org/1998/Math/MathML', 'msub'), ('http://www.w3.org/1999/xhtml', 'acronym'), ('http://www.w3.org/1998/Math/MathML', 'mtr'), ('http://www.w3.org/1999/xhtml', 'output'), ('http://www.w3.org/1999/xhtml', 'kbd'), ('http://www.w3.org/1999/xhtml', 'hr'), ('http://www.w3.org/2000/svg', 'font-face-src'), ('http://www.w3.org/1999/xhtml', 'tt'), ('http://www.w3.org/1999/xhtml', 'sub'), ('http://www.w3.org/1999/xhtml', 'dialog'), ('http://www.w3.org/2000/svg', 'polyline'), ('http://www.w3.org/2000/svg', 'path'), ('http://www.w3.org/1999/xhtml', 'time'), ('http://www.w3.org/1999/xhtml', 'tr'), ('http://www.w3.org/2000/svg', 'circle'), ('http://www.w3.org/1999/xhtml', 'span'), ('http://www.w3.org/1998/Math/MathML', 'mstyle'), ('http://www.w3.org/1999/xhtml', 'big'), ('http://www.w3.org/2000/svg', 'animateColor'), ('http://www.w3.org/2000/svg', 'missing-glyph'), ('http://www.w3.org/1999/xhtml', 'thead'), ('http://www.w3.org/1999/xhtml', 'keygen'), ('http://www.w3.org/1999/xhtml', 'datalist'), ('http://www.w3.org/1999/xhtml', 'spacer'), ('http://www.w3.org/1999/xhtml', 'dir'), ('http://www.w3.org/1999/xhtml', 'sup'), ('http://www.w3.org/1999/xhtml', 'img'), ('http://www.w3.org/2000/svg', 'text'), ('http://www.w3.org/1999/xhtml', 'center'), ('http://www.w3.org/1999/xhtml', 'blockquote'), ('http://www.w3.org/1999/xhtml', 'li'), ('http://www.w3.org/1999/xhtml', 'code'), ('http://www.w3.org/2000/svg', 'animate'), ('http://www.w3.org/1999/xhtml', 'audio'), ('http://www.w3.org/1999/xhtml', 'var'), ('http://www.w3.org/1999/xhtml', 'command'), ('http://www.w3.org/1999/xhtml', 'h3'), ('http://www.w3.org/1999/xhtml', 'div'), ('http://www.w3.org/1999/xhtml', 'ins'), ('http://www.w3.org/1999/xhtml', 'option'), ('http://www.w3.org/1998/Math/MathML', 'merror'), ('http://www.w3.org/2000/svg', 'marker'), ('http://www.w3.org/1999/xhtml', 'q'), ('http://www.w3.org/1998/Math/MathML', 'mo'), ('http://www.w3.org/1999/xhtml', 'strong'), ('http://www.w3.org/1999/xhtml', 'tbody'), ('http://www.w3.org/1999/xhtml', 'pre'), ('http://www.w3.org/2000/svg', 'set'), ('http://www.w3.org/1999/xhtml', 'sound'), ('http://www.w3.org/2000/svg', 'ellipse'), ('http://www.w3.org/1999/xhtml', 'h1'), ('http://www.w3.org/1999/xhtml', 'section'), ('http://www.w3.org/1999/xhtml', 'address'), ('http://www.w3.org/1999/xhtml', 'figure'), ('http://www.w3.org/1999/xhtml', 'th'), ('http://www.w3.org/1998/Math/MathML',* Sanitizes token stream of XHTML+MathML+SVG and of inline style attributes

__init__(source, allowed_elements=frozenset({('http://www.w3.org/1999/xhtml', 'h5'), ('http://www.w3.org/2000/svg', 'metadata'), ('http://www.w3.org/1998/Math/MathML', 'msub'), ('http://www.w3.org/1999/xhtml', 'acronym'), ('http://www.w3.org/1998/Math/MathML', 'mtr'), ('http://www.w3.org/1999/xhtml', 'output'), ('http://www.w3.org/1999/xhtml', 'kbd'), ('http://www.w3.org/1999/xhtml', 'hr'), ('http://www.w3.org/2000/svg', 'font-face-src'), ('http://www.w3.org/1999/xhtml', 'tt'), ('http://www.w3.org/1999/xhtml', 'sub'), ('http://www.w3.org/1999/xhtml', 'dialog'), ('http://www.w3.org/2000/svg', 'polyline'), ('http://www.w3.org/2000/svg', 'path'), ('http://www.w3.org/1999/xhtml', 'time'), ('http://www.w3.org/1999/xhtml', 'tr'), ('http://www.w3.org/2000/svg', 'circle'), ('http://www.w3.org/1999/xhtml', 'span'), ('http://www.w3.org/1998/Math/MathML', 'mstyle'), ('http://www.w3.org/1999/xhtml', 'big'), ('http://www.w3.org/2000/svg', 'animateColor'), ('http://www.w3.org/2000/svg', 'missing-glyph'), ('http://www.w3.org/1999/xhtml', 'thead'), ('http://www.w3.org/1999/xhtml', 'keygen'), ('http://www.w3.org/1999/xhtml', 'datalist'), ('http://www.w3.org/1999/xhtml', 'spacer'), ('http://www.w3.org/1999/xhtml', 'dir'), ('http://www.w3.org/1999/xhtml', 'sup'), ('http://www.w3.org/1999/xhtml', 'img'), ('http://www.w3.org/2000/svg', 'text'), ('http://www.w3.org/1999/xhtml', 'center'), ('http://www.w3.org/1999/xhtml', 'blockquote'), ('http://www.w3.org/1999/xhtml', 'li'), ('http://www.w3.org/1999/xhtml', 'code'), ('http://www.w3.org/2000/svg', 'animate'), ('http://www.w3.org/1999/xhtml', 'audio'), ('http://www.w3.org/1999/xhtml', 'var'), ('http://www.w3.org/1999/xhtml', 'command'), ('http://www.w3.org/1999/xhtml', 'h3'), ('http://www.w3.org/1999/xhtml', 'div'), ('http://www.w3.org/1999/xhtml', 'ins'), ('http://www.w3.org/1999/xhtml', 'option'), ('http://www.w3.org/1998/Math/MathML', 'merror'), ('http://www.w3.org/2000/svg', 'marker'), ('http://www.w3.org/1999/xhtml', 'q'), ('http://www.w3.org/1998/Math/MathML', 'mo'), ('http://www.w3.org/1999/xhtml', 'strong'), ('http://www.w3.org/1999/xhtml', 'tbody'), ('http://www.w3.org/1999/xhtml', 'pre'), ('http://www.w3.org/2000/svg', 'set'), ('http://www.w3.org/1999/xhtml', 'sound'), ('http://www.w3.org/2000/svg', 'ellipse'), ('http://www.w3.org/1999/xhtml', 'h1'), ('http://www.w3.org/1999/xhtml', 'section'), ('http://www.w3.org/1999/xhtml', 'address'), ('http://www.w3.org/1999/xhtml', 'figure'), ('http://www.w3.org/1999/xhtml', 'th'), ('http://www.w3.org/1998/Math/MathML', 'maction'), ('http://www.w3.org/1999/xhtml', 'footer'), ('http://www.w3.org/1999/xhtml', 'map'), ('http://www.w3.org/1998/Math/MathML', 'mtable'), ('http://www.w3.org/2000/svg', 'hkern'), ('http://www.w3.org/1998/Math/MathML', 'mover'), ('http://www.w3.org/1999/xhtml', 'multicol'), ('http://www.w3.org/1999/xhtml', 'progress'), ('http://www.w3.org/2000/svg', 'svg'), ('http://www.w3.org/1999/xhtml', 'a'), ('http://www.w3.org/1998/Math/MathML', 'munder'), ('http://www.w3.org/1999/xhtml', 'form'), ('http://www.w3.org/1999/xhtml', 'datagrid'), ('http://www.w3.org/1999/xhtml', 'samp'), ('http://www.w3.org/1999/xhtml', 'source'), ('http://www.w3.org/1999/xhtml', 'textarea'), ('http://www.w3.org/1999/xhtml', 'event-source'), ('http://www.w3.org/1999/xhtml', 'dt'), ('http://www.w3.org/2000/svg', 'radialGradient'), ('http://www.w3.org/1998/Math/MathML', 'mspace'), ('http://www.w3.org/1999/xhtml', 'i'), ('http://www.w3.org/2000/svg', 'font-face'), ('http://www.w3.org/1999/xhtml', 's'), ('http://www.w3.org/1999/xhtml', 'ol'), ('http://www.w3.org/2000/svg', 'switch'), ('http://www.w3.org/2000/svg', 'a'), ('http://www.w3.org/1998/Math/MathML', 'mpadded'), ('http://www.w3.org/2000/svg', 'tspan'), ('http://www.w3.org/1998/Math/MathML', 'msqrt'), ('http://www.w3.org/1999/xhtml', 'meter'), ('http://www.w3.org/2000/svg', 'line'), ('http://www.w3.org/1998/Math/MathML', 'mtd'), ('http://www.w3.org/1999/xhtml', 'button'), ('http://www.w3.org/2000/svg', 'desc'), ('http://www.w3.org/2000/svg', 'glyph'), ('http://www.w3.org/1998/Math/MathML', 'mn'), ('http://www.w3.org/1999/xhtml', 'font'), ('http://www.w3.org/2000/svg', 'linearGradient'), ('http://www.w3.org/2000/svg', 'mpath'), ('http://www.w3.org/1999/xhtml', 'u'), ('http://www.w3.org/1999/xhtml', 'fieldset'), ('http://www.w3.org/2000/svg', 'title'), ('http://www.w3.org/1998/Math/MathML', 'mtext'), ('http://www.w3.org/1999/xhtml', 'br'), ('http://www.w3.org/1999/xhtml', 'dd'), ('http://www.w3.org/1999/xhtml', 'strike'), ('http://www.w3.org/1998/Math/MathML', 'mprescripts'), ('http://www.w3.org/1999/xhtml', 'dfn'), ('http://www.w3.org/2000/svg', 'g'), ('http://www.w3.org/1999/xhtml', 'nav'), ('http://www.w3.org/1999/xhtml', 'details'), ('http://www.w3.org/1999/xhtml', 'ul'), ('http://www.w3.org/1999/xhtml', 'optgroup'), ('http://www.w3.org/2000/svg', 'animateTransform'), ('http://www.w3.org/1998/Math/MathML', 'munderover'),

Creates a Filter

Parameters

- **allowed_elements** – set of elements to allow–everything else will be escaped
- **allowed_attributes** – set of attributes to allow in elements–everything else will be stripped
- **allowed_css_properties** – set of CSS properties to allow–everything else will be stripped
- **allowed_css_keywords** – set of CSS keywords to allow–everything else will be stripped
- **allowed_svg_properties** – set of SVG properties to allow–everything else will be removed
- **allowed_protocols** – set of allowed protocols for URIs
- **allowed_content_types** – set of allowed content types for `data` URIs.
- **attr_val_is_uri** – set of attributes that have URI values–values that have a scheme not listed in `allowed_protocols` are removed
- **svg_attr_val_allows_ref** – set of SVG attributes that can have references
- **svg_allow_local_href** – set of SVG elements that can have local hrefs–these are removed

`whitespace` Module

class `html5lib.filters.whitespace.`**`Filter`**(*source*)

Bases: `html5lib.filters.base.Filter`

Collapses whitespace except in pre, textarea, and script elements

treebuilders Package

`treebuilders` Package

A collection of modules for building different kinds of trees from HTML documents.

To create a treebuilder for a new type of tree, you need to do implement several things:

1. A set of classes for various types of elements: Document, Doctype, Comment, Element. These must implement the interface of `base.treebuilders.Node` (although comment nodes have a different signature for their constructor, see `treebuilders.etree.Comment`) Textual content may also be implemented as another node type, or not, as your tree implementation requires.
2. A treebuilder object (called `TreeBuilder` by convention) that inherits from `treebuilders.base.TreeBuilder`. This has 4 required attributes:
   - `documentClass` - the class to use for the bottommost node of a document
   - `elementClass` - the class to use for HTML Elements
   - `commentClass` - the class to use for comments
   - `doctypeClass` - the class to use for doctypes

   It also has one required method:

- `getDocument` - Returns the root node of the complete document tree

3. If you wish to run the unit tests, you must also create a `testSerializer` method on your treebuilder which accepts a node and returns a string containing Node and its children serialized according to the format used in the unittests

`html5lib.treebuilders.`**`getTreeBuilder`**(*treeType*, *implementation=None*, ***kwargs*)

Get a TreeBuilder class for various types of trees with built-in support

Parameters

- **`treeType`** – the name of the tree type required (case-insensitive). Supported values are:
  - ”dom” - A generic builder for DOM implementations, defaulting to a xml.dom.minidom based implementation.
  - ”etree” - A generic builder for tree implementations exposing an ElementTree-like interface, defaulting to xml.etree.cElementTree if available and xml.etree.ElementTree if not.
  - ”lxml” - A etree-based builder for lxml.etree, handling limitations of lxml’s implementation.
- **`implementation`** – (Currently applies to the “etree” and “dom” tree types). A module implementing the tree type e.g. xml.etree.ElementTree or xml.etree.cElementTree.
- **`kwargs`** – Any additional options to pass to the TreeBuilder when creating it.

Example:

```
>>> from html5lib.treebuilders import getTreeBuilder
>>> builder = getTreeBuilder('etree')
```

`base` Module

`class` `html5lib.treebuilders.base.`**`Node`**(*name*)

Bases: `object`

Represents an item in the tree

`__init__`(*name*)

Creates a Node

Parameters **`name`** – The tag name associated with the node

`appendChild`(*node*)

Insert node as a child of the current node

Parameters **`node`** – the node to insert

`cloneNode`()

Return a shallow copy of the current node i.e. a node with the same name and attributes but with no parent or child nodes

`hasContent`()

Return true if the node has children or text, false otherwise

`insertBefore`(*node*, *refNode*)

Insert node as a child of the current node, before refNode in the list of child nodes. Raises ValueError if refNode is not a child of the current node

Parameters

- **node** – the node to insert
- **refNode** – the child node to insert the node before

insertText (*data*, *insertBefore=None*)

Insert data as text in the current node, positioned before the start of node insertBefore or to the end of the node's text.

Parameters

- **data** – the data to insert
- **insertBefore** – True if you want to insert the text before the node and False if you want to insert it after the node

removeChild (*node*)

Remove node from the children of the current node

Parameters node – the child node to remove

reparentChildren (*newParent*)

Move all the children of the current node to newParent. This is needed so that trees that don't store text as nodes move the text in the correct way

Parameters newParent – the node to move all this node's children to

class html5lib.treebuilders.base.**TreeBuilder** (*namespaceHTMLElements*)

Bases: `object`

Base treebuilder implementation

- documentClass - the class to use for the bottommost node of a document
- elementClass - the class to use for HTML Elements
- commentClass - the class to use for comments
- doctypeClass - the class to use for doctypes

__init__ (*namespaceHTMLElements*)

Create a TreeBuilder

Parameters namespaceHTMLElements – whether or not to namespace HTML elements

createElement (*token*)

Create an element but don't insert it anywhere

elementInActiveFormattingElements (*name*)

Check if an element exists between the end of the active formatting elements and the last marker. If it does, return it, else return false

getDocument ()

Return the final tree

getFragment ()

Return the final fragment

getTableMisnestedNodePosition ()

Get the foster parent element, and sibling to insert before (or None) when inserting a misnested table node

insertElementTable (*token*)

Create an element and insert it into the tree

insertText (*data*, *parent=None*)

Insert text data.

testSerializer(*node*)

Serialize the subtree of node in the format required by unit tests

Parameters **node** – the node from which to start serializing

dom Module

etree Module

etree_lxml Module

Module for supporting the lxml.etree library. The idea here is to use as much of the native library as possible, without using fragile hacks like custom element names that break between releases. The downside of this is that we cannot represent all possible trees; specifically the following are known to cause problems:

Text or comments as siblings of the root element Docypes with no name

When any of these things occur, we emit a DataLossWarning

html5lib.treebuilders.etree_lxml.**tostring**(*element*)

Serialize an element and its child nodes to a string

treewalkers Package

treewalkers Package

A collection of modules for iterating through different kinds of tree, generating tokens identical to those produced by the tokenizer module.

To create a tree walker for a new type of tree, you need to implement a tree walker object (called TreeWalker by convention) that implements a 'serialize' method which takes a tree as sole argument and returns an iterator which generates tokens.

html5lib.treewalkers.**getTreeWalker**(*treeType*, *implementation=None*, ***kwargs*)

Get a TreeWalker class for various types of tree with built-in support

Parameters

- **treeType** (*str*) – the name of the tree type required (case-insensitive). Supported values are:
  - ”dom”: The xml.dom.minidom DOM implementation
  - ”etree”: A generic walker for tree implementations exposing an elementtree-like interface (known to work with ElementTree, cElementTree and lxml.etree).
  - ”lxml”: Optimized walker for lxml.etree
  - ”genshi”: a Genshi stream
- **implementation** – A module implementing the tree type e.g. xml.etree.ElementTree or cElementTree (Currently applies to the “etree” tree type only).
- **kwargs** – keyword arguments passed to the etree walker–for other walkers, this has no effect

Returns a TreeWalker class

`html5lib.treewalkers.`**`pprint`**(*walker*)

Pretty printer for tree walkers

Takes a TreeWalker instance and pretty prints the output of walking the tree.

Parameters **`walker`** – a TreeWalker instance

`base` Module

`class` `html5lib.treewalkers.base.`**`TreeWalker`**(*tree*)

Bases: `object`

Walks a tree yielding tokens

Tokens are dicts that all have a `type` field specifying the type of the token.

`__init__`(*tree*)

Creates a TreeWalker

Parameters **`tree`** – the tree to walk

`comment`(*data*)

Generates a Comment token

Parameters **`data`** – the comment

Returns Comment token

`doctype`(*name*, *publicId=None*, *systemId=None*)

Generates a Doctype token

Parameters

- **`name`** –
- **`publicId`** –
- **`systemId`** –

Returns the Doctype token

`emptyTag`(*namespace*, *name*, *attrs*, *hasChildren=False*)

Generates an EmptyTag token

Parameters

- **`namespace`** – the namespace of the token–can be `None`
- **`name`** – the name of the element
- **`attrs`** – the attributes of the element as a dict
- **`hasChildren`** – whether or not to yield a SerializationError because this tag shouldn't have children

Returns EmptyTag token

`endTag`(*namespace*, *name*)

Generates an EndTag token

Parameters

- **`namespace`** – the namespace of the token–can be `None`
- **`name`** – the name of the element

Returns EndTag token

entity(*name*)

Generates an Entity token

Parameters **name** – the entity name

Returns an Entity token

error(*msg*)

Generates an error token with the given message

Parameters **msg** – the error message

Returns SerializeError token

startTag(*namespace*, *name*, *attrs*)

Generates a StartTag token

Parameters

- **namespace** – the namespace of the token–can be `None`
- **name** – the name of the element
- **attrs** – the attributes of the element as a dict

Returns StartTag token

text(*data*)

Generates SpaceCharacters and Characters tokens

Depending on what's in the data, this generates one or more `SpaceCharacters` and `Characters` tokens.

For example:

```
>>> from html5lib.treewalkers.base import TreeWalker
>>> # Give it an empty tree just so it instantiates
>>> walker = TreeWalker([])
>>> list(walker.text(''))
[]
>>> list(walker.text('  '))
[{u'data': '  ', u'type': u'SpaceCharacters'}]
>>> list(walker.text(' abc '))
[{u'data': ' ', u'type': u'SpaceCharacters'},
{u'data': u'abc', u'type': u'Characters'},
{u'data': u' ', u'type': u'SpaceCharacters'}]
```

Parameters **data** – the text data

Returns one or more `SpaceCharacters` and `Characters` tokens

unknown(*nodeType*)

Handles unknown node types

`dom` Module

`etree` Module

`etree_lxml` Module

`genshi` Module

treeadapters Package

`treeadapters` Package

Tree adapters let you convert from one tree structure to another

Example:

```
import html5lib
from html5lib.treeadapters import genshi

doc = '<html><body>Hi!</body></html>'
treebuilder = html5lib.getTreeBuilder('etree')
parser = html5lib.HTMLParser(tree=treebuilder)
tree = parser.parse(doc)
TreeWalker = html5lib.getTreeWalker('etree')

genshi_tree = genshi.to_genshi(TreeWalker(tree))
```

`html5lib.treeadapters.genshi.`**`to_genshi`**`(`*walker*`)`

Convert a tree to a genshi tree

Parameters **`walker`** – the treewalker to use to walk the tree to convert it

Returns generator of genshi nodes

`html5lib.treeadapters.sax.`**`to_sax`**`(`*walker*, *handler*`)`

Call SAX-like content handler based on treewalker walker

Parameters

- **`walker`** – the treewalker to use to walk the tree to convert it
- **`handler`** – SAX handler to use

6.3 Change Log

6.3.1 1.0.1

Released on December 7, 2017

Breaking changes:

- Drop support for Python 2.6. (#330) (Thank you, Hugo, Will Kahn-Greene!)
- Remove `utils/spider.py` (#353) (Thank you, Jon Dufresne!)

Features:

- Improve documentation. (#300, #307) (Thank you, Jon Dufresne, Tom Most, Will Kahn-Greene!)
- Add iframe seamless boolean attribute. (Thank you, Ritwik Gupta!)
- Add itemscope as a boolean attribute. (#194) (Thank you, Jonathan Vanasco!)
- Support Python 3.6. (#333) (Thank you, Jon Dufresne!)
- Add CI support for Windows using AppVeyor. (Thank you, John Vandenberg!)
- Improve testing and CI and add code coverage (#323, #334), (Thank you, Jon Dufresne, John Vandenberg, Geoffrey Sneddon, Will Kahn-Greene!)
- Semver-compliant version number.

Bug fixes:

- Add support for setuptools < 18.5 to support environment markers. (Thank you, John Vandenberg!)
- Add explicit dependency for six >= 1.9. (Thank you, Eric Amorde!)
- Fix regexes to work with Python 3.7 regex adjustments. (#318, #379) (Thank you, Benedikt Morbach, Ville Skyttä, Mark Vasilkov!)
- Fix alphabeticalattributes filter namespace bug. (#324) (Thank you, Will Kahn-Greene!)
- Include license file in generated wheel package. (#350) (Thank you, Jon Dufresne!)
- Fix annotation-xml typo. (#339) (Thank you, Will Kahn-Greene!)
- Allow uppercase hex chararcters in CSS colour check. (#377) (Thank you, Komal Dembla, Hugo!)

6.3.2 1.0

Released and unreleased on December 7, 2017. Badly packaged release.

6.3.3 0.999999999/1.0b10

Released on July 15, 2016

- Fix attribute order going to the tree builder to be document order instead of reverse document order(!).

6.3.4 0.99999999/1.0b9

Released on July 14, 2016

- **Added ordereddict as a mandatory dependency on Python 2.6.**
- Added `lxml`, `genshi`, `datrie`, `charade`, and `all` extras that will do the right thing based on the specific interpreter implementation.
- Now requires the `mock` package for the testsuite.
- Cease supporting DATrie under PyPy.
- **Remove PullDOM support, as this hasn't ever been properly tested, doesn't entirely work, and as far as I can tell is completely unused by anyone.**
- Move testsuite to `py.test`.
- **Fix #124: move to webencodings for decoding the input byte stream; this makes html5lib compliant with the Encoding Standard, and introduces a required dependency on webencodings.**

- **Cease supporting Python 3.2 (in both CPython and PyPy forms).**
- **Fix comments containing double-dash with lxml 3.5 and above.**
- **Use scripting disabled by default (as we don't implement scripting).**
- **Fix #11, avoiding the XSS bug potentially caused by serializer allowing attribute values to be escaped out of in old browser versions, changing the quote_attr_values option on serializer to take one of three values, "always" (the old True value), "legacy" (the new option, and the new default), and "spec" (the old False value, and the old default).**
- **Fix #72 by rewriting the sanitizer to apply only to treewalkers (instead of the tokenizer); as such, this will require amending all callers of it to use it via the treewalker API.**
- **Drop support of charade, now that chardet is supported once more.**
- **Replace the charset keyword argument on parse and related methods with a set of keyword arguments: override_encoding, transport_encoding, same_origin_parent_encoding, likely_encoding, and default_encoding.**
- **Move filters._base, treebuilder._base, and treewalkers._base to .base to clarify their status as public.**
- **Get rid of the sanitizer package. Merge sanitizer.sanitize into the sanitizer.htmlsanitizer module and move that to sanitizer. This means anyone who used sanitizer.sanitize or sanitizer.HTMLSanitizer needs no code changes.**
- **Rename treewalkers.lxmletree to .etree_lxml and treewalkers.genshistream to .genshi to have a consistent API.**
- Move a whole load of stuff (inputstream, ihatexml, trie, tokenizer, utils) to be underscore prefixed to clarify their status as private.

6.3.5 0.9999999/1.0b8

Released on September 10, 2015

- Fix #195: fix the sanitizer to drop broken URLs (it threw an exception between 0.9999 and 0.999999).

6.3.6 0.999999/1.0b7

Released on July 7, 2015

- Fix #189: fix the sanitizer to allow relative URLs again (as it did prior to 0.9999/1.0b5).

6.3.7 0.99999/1.0b6

Released on April 30, 2015

- Fix #188: fix the sanitizer to not throw an exception when sanitizing bogus data URLs.

6.3.8 0.9999/1.0b5

Released on April 29, 2015

- Fix #153: Sanitizer fails to treat some attributes as URLs. Despite how this sounds, this has no known security implications. No known version of IE (5.5 to current), Firefox (3 to current), Safari (6 to current), Chrome (1 to current), or Opera (12 to current) will run any script provided in these attributes.

- Pass error message to the ParseError exception in strict parsing mode.
- Allow data URIs in the sanitizer, with a whitelist of content-types.
- Add support for Python implementations that don't support lone surrogates (read: Jython). Fixes #2.
- Remove localization of error messages. This functionality was totally unused (and untested that everything was localizable), so we may as well follow numerous browsers in not supporting translating technical strings.
- Expose treewalkers.pprint as a public API.
- Add a documentEncoding property to HTML5Parser, fix #121.

6.3.9 0.999

Released on December 23, 2013

- Fix #127: add work-around for CPython issue #20007: .read(0) on http.client.HTTPResponse drops the rest of the content.
- Fix #115: lxml treewalker can now deal with fragments containing, at their root level, text nodes with non-ASCII characters on Python 2.

6.3.10 0.99

Released on September 10, 2013

- No library changes from 1.0b3; released as 0.99 as pip has changed behaviour from 1.4 to avoid installing pre-release versions per PEP 440.

6.3.11 1.0b3

Released on July 24, 2013

- Removed `RecursiveTreeWalker` from `treewalkers._base`. Any implementation using it should be moved to `NonRecursiveTreeWalker`, as everything bundled with html5lib has for years.
- Fix #67 so that `BufferedStream` to correctly returns a bytes object, thereby fixing any case where html5lib is passed a non-seekable RawIOBase-like object.

6.3.12 1.0b2

Released on June 27, 2013

- Removed reordering of attributes within the serializer. There is now an `alphabetical_attributes` option which preserves the previous behaviour through a new filter. This allows attribute order to be preserved through html5lib if the tree builder preserves order.
- Removed `dom2sax` from DOM treebuilders. It has been replaced by `treeadapters.sax.to_sax` which is generic and supports any treewalker; it also resolves all known bugs with `dom2sax`.
- Fix treewalker assertions on hitting bytes strings on Python 2. Previous to 1.0b1, treewalkers coped with mixed bytes/unicode data on Python 2; this reintroduces this prior behaviour on Python 2. Behaviour is unchanged on Python 3.

6.3.13 1.0b1

Released on May 17, 2013

- Implementation updated to implement the HTML specification as of 5th May 2013 (SVN revision r7867).
- Python 3.2+ supported in a single codebase using the `six` library.
- Removed support for Python 2.5 and older.
- Removed the deprecated Beautiful Soup 3 treebuilder. `beautifulsoup4` can use `html5lib` as a parser instead. Note that since it doesn't support namespaces, foreign content like SVG and MathML is parsed incorrectly.
- Removed `simpletree` from the package. The default tree builder is now `etree` (using the `xml.etree.cElementTree` implementation if available, and `xml.etree.ElementTree` otherwise).
- Removed the `XHTMLSerializer` as it never actually guaranteed its output was well-formed XML, and hence provided little of use.
- Removed default DOM treebuilder, so `html5lib.treebuilders.dom` is no longer supported. `html5lib.treebuilders.getTreeBuilder("dom")` will return the default DOM treebuilder, which uses `xml.dom.minidom`.
- Optional heuristic character encoding detection now based on `charade` for Python 2.6 - 3.3 compatibility.
- Optional `Genshi` treewalker support fixed.
- Many bugfixes, including:
  - #33: null in attribute value breaks XML AttValue;
  - #4: nested, indirect descendant, <button> causes infinite loop;
  - Google Code 215: Properly detect seekable streams;
  - Google Code 206: add support for <video preload=...>, <audio preload=...>;
  - Google Code 205: add support for <video poster=...>;
  - Google Code 202: Unicode file breaks InputStream.
- Source code is now mostly PEP 8 compliant.
- Test harness has been improved and now depends on `nose`.
- Documentation updated and moved to https://html5lib.readthedocs.io/.

6.3.14 0.95

Released on February 11, 2012

6.3.15 0.90

Released on January 17, 2010

6.3.16 0.11.1

Released on June 12, 2008

6.3.17 0.11

Released on June 10, 2008

6.3.18 0.10

Released on October 7, 2007

6.3.19 0.9

Released on March 11, 2007

6.3.20 0.2

Released on January 8, 2007

6.4 License

Copyright (c) 2006-2013 James Graham and other contributors

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index