lazyxml Documentation

Release 1.3.0

Ryan Fau

Dec 10, 2019

Contents

CHAPTER 1

API Documentation

1.1 `lazyxml` – A simple xml parse and build lib

1.1.1 `loads()` – Load xml content to python object.

A simple xml parse and build library.

`lazyxml.`**`loads`**(*content*, *encoding=None*, *unescape=False*, *strip_root=True*, *strip_attr=True*, *strip=True*, *errors='strict'*)

Load xml content to python object.

```
>>> import lazyxml
```

```
>>> xml = '<demo><foo>foo</foo><bar>bar</bar></demo>'
>>> lazyxml.loads(xml)
{'bar': 'bar', 'foo': 'foo'}
```

```
>>> xml = '<demo><foo>foo</foo><bar>bar</bar></demo>'
>>> lazyxml.loads(xml, strip_root=False)
{'demo': {'bar': 'bar', 'foo': 'foo'}}
```

```
>>> xml = '<demo><foo>foo</foo><bar>1</bar><bar>2</bar></demo>'
>>> lazyxml.loads(xml)
{'bar': ['1', '2'], 'foo': 'foo'}
```

```
>>> xml = '<root xmlns:h="http://www.w3.org/TR/html4/">&lt;demo&gt;&lt;foo&gt;foo&
↪lt;/foo&gt;&lt;bar&gt;bar&lt;/bar&gt;&lt;/demo&gt;</root>'
>>> lazyxml.loads(xml, unescape=True, strip_root=False)
{'root': {'demo': {'bar': 'bar', 'foo': 'foo'}}}
```

Parameters

- **`content`** (*str*) – xml content.

- **encoding** (*str*) – xml content encoding. if not set, will guess from xml header declare if possible.
- **unescape** (*bool*) – whether to unescape xml html entity character. Default to `False`.
- **strip_root** (*bool*) – whether to strip root. Default to `True`.
- **strip_attr** (*bool*) – whether to strip tag attrs. Default to `True`.
- **strip** (*bool*) – whether to strip whitespace. Default to `True`.
- **errors** (*string*) – the xml content decode error handling scheme. Default to `strict`.

Return type dict

Changed in version 1.2.1: The `strip_attr` option supported to decide whether return the element attributes for parse result.

1.1.2 `load()` – Load xml content from file and convert to python object.

A simple xml parse and build library.

`lazyxml.`**load**(*fp*, *encoding=None*, *unescape=False*, *strip_root=True*, *strip_attr=True*, *strip=True*, *errors='strict'*)

Load xml content from file and convert to python object.

```
>>> import lazyxml
>>> with open('demo.xml', 'rb') as fp:
>>>     lazyxml.load(fp)
```

```
>>> from cStringIO import StringIO
>>> buf = StringIO('<?xml version="1.0" encoding="utf-8"?><demo><foo><![CDATA[
↪<foo>]]></foo><bar><![CDATA[1]]></bar><bar><![CDATA[2]]></bar></demo>')
>>> lazyxml.load(buf)
{'bar': ['1', '2'], 'foo': '<foo>'}
>>> buf.close()
```

Parameters

- **fp** – a file or file-like object that support `.read()` to read the xml content
- **encoding** (*str*) – xml content encoding. if not set, will guess from xml header declare if possible.
- **unescape** (*bool*) – whether to unescape xml html entity character. Default to `False`.
- **strip_root** (*bool*) – whether to strip root. Default to `True`.
- **strip_attr** (*bool*) – whether to strip tag attrs. Default to `True`.
- **strip** (*bool*) – whether to strip whitespace. Default to `True`.
- **errors** (*string*) – the xml content decode error handling scheme. Default to `strict`.

Return type dict

Changed in version 1.2.1: The `strip_attr` option supported to decide whether return the element attributes for parse result.

1.1.3 `dumps()` – Dump python object to xml.

A simple xml parse and build library.

`lazyxml.`**`dumps`**(*obj*, *encoding=None*, *header_declare=True*, *version=None*, *root=None*, *cdata=True*, *indent=None*, *ksort=False*, *reverse=False*, *errors='strict'*, *hasattr=False*, *attrkey=None*, *valuekey=None*)

Dump python object to xml.

```
>>> import lazyxml
```

```
>>> data = {'demo': {'foo': '<foo>', 'bar': ['1', '2']}}
```

```
>>> lazyxml.dumps(data)
'<?xml version="1.0" encoding="utf-8"?><demo><foo><![CDATA[<foo>]]></foo><bar><!
↪[CDATA[1]]></bar><bar><![CDATA[2]]></bar></demo>'
```

```
>>> lazyxml.dumps(data, header_declare=False)
'<demo><foo><![CDATA[<foo>]]></foo><bar><![CDATA[1]]></bar><bar><![CDATA[2]]></
↪bar></demo>'
```

```
>>> lazyxml.dumps(data, cdata=False)
'<?xml version="1.0" encoding="utf-8"?><demo><foo>&lt;foo&gt;</foo><bar>1</bar>
↪<bar>2</bar></demo>'
```

```
>>> print lazyxml.dumps(data, indent=' ' * 4)
<?xml version="1.0" encoding="utf-8"?>
<demo>
    <foo><![CDATA[<foo>]]></foo>
    <bar><![CDATA[1]]></bar>
    <bar><![CDATA[2]]></bar>
</demo>
```

```
>>> lazyxml.dumps(data, ksort=True)
'<?xml version="1.0" encoding="utf-8"?><demo><bar><![CDATA[1]]></bar><bar><!
↪[CDATA[2]]></bar><foo><![CDATA[<foo>]]></foo></demo>'
```

```
>>> lazyxml.dumps(data, ksort=True, reverse=True)
'<?xml version="1.0" encoding="utf-8"?><demo><foo><![CDATA[<foo>]]></foo><bar><!
↪[CDATA[1]]></bar><bar><![CDATA[2]]></bar></demo>'
```

Note: Data that has attributes convert to xml see `demo/dump.py`.

Parameters

- **`obj`** – data for dump to xml.
- **`encoding`** (*`str`*) – xml content encoding. if not set, `consts.Default.ENCODING` used.
- **`header_declare`** (*`bool`*) – declare xml header. Default to `True`.
- **`version`** (*`str`*) – xml version. if not set, `consts.Default.VERSION` used.
- **`root`** (*`str`*) – xml root. Default to `None`.

- **cdata** (*bool*) – use cdata. Default to `True`.
- **indent** (*str*) – xml pretty indent. Default to `None`.
- **ksort** (*bool*) – sort xml element keys. Default to `False`.
- **reverse** (*bool*) – sort xml element keys but reverse. Default to `False`.
- **errors** (*str*) – xml content decode error handling scheme. Default to `strict`.
- **hasattr** (*bool*) – data element has attributes. Default to `False`.
- **attrkey** (*str*) – element tag attribute identification. if not set, `consts.Default.KEY_ATTR` used.
- **valuekey** (*str*) – element tag value identification. if not set, `consts.Default.KEY_VALUE` used.

Return type str

1.1.4 `dump()` – Dump python object to file.

A simple xml parse and build library.

`lazyxml.`**dump**(*obj*, *fp*, *encoding=None*, *header_declare=True*, *version=None*, *root=None*, *cdata=True*, *indent=None*, *ksort=False*, *reverse=False*, *errors='strict'*, *hasattr=False*, *attrkey=None*, *valuekey=None*)

Dump python object to file.

```
>>> import lazyxml
>>> data = {'demo': {'foo': 1, 'bar': 2}}
>>> lazyxml.dump(data, 'dump.xml')
>>> with open('dump-fp.xml', 'w') as fp:
>>>     lazyxml.dump(data, fp)
```

```
>>> from cStringIO import StringIO
>>> data = {'demo': {'foo': 1, 'bar': 2}}
>>> buf = StringIO()
>>> lazyxml.dump(data, buf)
>>> buf.getvalue()
<?xml version="1.0" encoding="utf-8"?><demo><foo><![CDATA[1]]></foo><bar><!
↪[CDATA[2]]></bar></demo>
>>> buf.close()
```

Parameters

- **obj** – data for dump to xml.
- **fp** – a filename or a file or file-like object that support `.write()` to write the xml content.
- **encoding** (*str*) – xml content encoding. if not set, `consts.Default.ENCODING` used.
- **header_declare** (*bool*) – declare xml header. Default to `True`.
- **version** (*str*) – xml version. if not set, `consts.Default.VERSION` used.
- **root** (*str*) – xml root. Default to `None`.
- **cdata** (*bool*) – use cdata. Default to `True`.
- **indent** (*str*) – xml pretty indent. Default to `None`.

- **ksort** (*bool*) – sort xml element keys. Default to `False`.
- **reverse** (*bool*) – sort xml element keys but reverse. Default to `False`.
- **errors** (*str*) – xml content decode error handling scheme. Default to `strict`.
- **hasattr** (*bool*) – data element has attributes. Default to `False`.
- **attrkey** (*str*) – element tag attribute identification. if not set, `consts.Default.KEY_ATTR` used.
- **valuekey** (*str*) – element tag value identification. if not set, `consts.Default.KEY_VALUE` used.

Changed in version 1.2: The *fp* is a filename of string before this. It can now be a file or file-like object that support `.write()` to write the xml content.

1.1.5 `builder` – XML Builder Module

class lazyxml.builder.**Builder**(*encoding=None*, *header_declare=True*, *version=None*, *root=None*, *cdata=True*, *indent=None*, *ksort=False*, *reverse=False*, *errors='strict'*, *hasattr=False*, *attrkey=None*, *valuekey=None*)

Simple xml builder

dict2xml(*data*)

Convert dict to xml.

> **Warning:** **DEPRECATED:** *dict2xml()* is deprecated. Please use *object2xml()* instead.

Deprecated since version 1.2.

object2xml(*data*)

Convert python object to xml string.

Parameters **data** – data for build xml. If don't provide the `root` option, type of `data` must be dict and `len(data) == 1`.

Return type str or unicode

New in version 1.2.

static build_xml_header(*encoding=None*, *version=None*)

Build xml header include version and encoding.

build_tree(*data*, *tagname*, *attrs=None*, *depth=0*)

Build xml tree.

Parameters

- **data** – data for build xml.
- **tagname** – element tag name.
- **attrs** (*dict or None*) – element attributes. Default`None`.
- **depth** (*int*) – element depth of the hierarchy. Default`0`.

check_structure(*keys*)

Check structure availability by `attrkey` and `valuekey` option.

pickdata(*data*)
Pick data from `attrkey` and `valuekey` option.

Returns a pair of (attrs, values)

Return type tuple

static safedata(*data*, *cdata=True*)
Convert xml special chars to entities.

Parameters

- **data** – the data will be converted safe.
- **cdata** (*bool*) – whether to use cdata. Default`True`. If not, use `cgi.escape()` to convert data.

Return type str

classmethod build_tag(*tag*, *text=''*, *attrs=None*)
Build tag full info include the attributes.

Parameters

- **tag** – tag name.
- **text** – tag text.
- **attrs** (*dict or None*) – tag attributes. Default`None`.

Return type str

static build_attr(*attrs*)
Build tag attributes.

Parameters **attrs** (*dict*) – tag attributes

Return type str

classmethod tag_start(*tag*, *attrs=None*)
Build started tag info.

Parameters

- **tag** – tag name
- **attrs** (*dict or None*) – tag attributes. Default`None`.

Return type str

static tag_end(*tag*)
Build closed tag info.

Parameters **tag** – tag name

Return type str

1.1.6 `parser` – XML Parser Module

class lazyxml.parser.**Parser**(*encoding=None*, *unescape=False*, *strip_root=True*, *strip_attr=True*, *strip=True*, *errors='strict'*)
Simple xml parser

xml2dict(*content*)
Convert xml content to dict.

> **Warning:** **DEPRECATED:** *xml2dict()* is deprecated. Please use *xml2object()* instead.

Deprecated since version 1.2.

xml2object(*content*)
Convert xml content to python object.

Parameters **content** – xml content

Return type dict

New in version 1.2.

xml_filter(*content*)
Filter and preprocess xml content

Parameters **content** – xml content

Return type str

static guess_xml_encoding(*content*)
Guess encoding from xml header declaration.

Parameters **content** – xml content

Return type str or None

static strip_xml_header(*content*)
Strip xml header

Parameters **content** – xml content

Return type str

classmethod parse(*element*)
Parse xml element.

Parameters **element** – an Element instance

Return type dict

classmethod parse_full(*element*)
Parse xml element include the node attributes.

Parameters **element** – an Element instance

Return type dict

New in version 1.2.1.

classmethod get_node(*element*)
Get node info.

Parse element and get the element tag info. Include tag name, value, attribute, namespace.

Parameters **element** – an Element instance

Return type dict

static split_namespace(*tag*)
Split tag namespace.

Parameters **tag** – tag name

Returns a pair of (namespace, tag)

Return type tuple

CHAPTER 2

Changes

See the changelog for a full list of changes to *lazyxml*.

CHAPTER 3

About This Documentation

This documentation is generated using the Sphinx documentation generator. The source files for the documentation are located in the *docs/* directory of the **lazyxml** distribution. To generate the docs locally run the following command from the *docs/* directory of the **lazyxml** source:

```
$ cd docs
$ make html
```

or use `make help` to generate other format.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

l

Index