

NLTK

Documentation

nltk.tokenize package

Submodules

- `nltk.tokenize.api` module
- `nltk.tokenize.casual` module
- `nltk.tokenize.destructive` module
- `nltk.tokenize.legality_principle` module
- `nltk.tokenize.mwe` module
- `nltk.tokenize.nist` module
- `nltk.tokenize.punkt` module
- `nltk.tokenize.regex` module
- `nltk.tokenize.repp` module
- `nltk.tokenize.sexpr` module
- `nltk.tokenize.simple` module
- `nltk.tokenize.sonority_sequencing` module
- `nltk.tokenize.stanford` module
- `nltk.tokenize.stanford_segmenter` module
- `nltk.tokenize.texttiling` module
- `nltk.tokenize.toktok` module
- `nltk.tokenize.treebank` module
- `nltk.tokenize.util` module

Search

NLTK Documentation

API Reference

Example Usage

Module Index

Wiki

FAQ

Open Issues

NLTK on GitHub

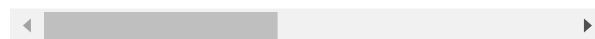
[Installation](#)[Installing NLTK](#)[Installing NLTK Data](#)[More](#)[Release Notes](#)[Contributing to NLTK](#)[NLTK Team](#)

Module contents

NLTK Tokenizer Package

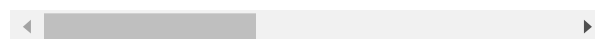
Tokenizers divide strings into lists of substrings. For example, tokenizers can be used to find the words and punctuation in a string:

```
>>> from nltk.tokenize import  
>>> s = '''Good muffins cost  
... two of them.\n\nThanks.''  
>>> word_tokenize(s)  
['Good', 'muffins', 'cost', '  
'Please', 'buy', 'me', 'two',
```



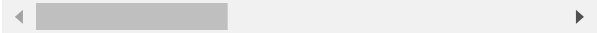
This particular tokenizer requires the Punkt sentence tokenization models to be installed. NLTK also provides a simpler, regular-expression based tokenizer, which splits text on whitespace and punctuation:

```
>>> from nltk.tokenize import  
>>> wordpunct_tokenize(s)  
['Good', 'muffins', 'cost', '  
'Please', 'buy', 'me', 'two',
```



We can also operate at the level of sentences, using the sentence tokenizer directly as follows:

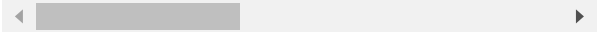
```
>>> from nltk.tokenize import  
>>> sent_tokenize(s)  
['Good muffins cost $3.88\nin  
>>> [word_tokenize(t) for t i  
[['Good', 'muffins', 'cost',  
['Please', 'buy', 'me', 'two']
```



Caution: when tokenizing a Unicode string, make sure you are not using an encoded version of the string (it may be necessary to decode it first, e.g. with `s.decode("utf8")`).

NLTK tokenizers can produce token-spans, represented as tuples of integers having the same semantics as string slices, to support efficient comparison of tokenizers. (These methods are implemented as generators.)

```
>>> from nltk.tokenize import  
>>> list(WhitespaceTokenizer(  
[(0, 4), (5, 12), (13, 17), (  
(45, 48), (49, 51), (52, 55),
```



There are numerous ways to tokenize text. If you need more control over tokenization, see the other methods provided in this package.

For further information, please see Chapter 3 of the NLTK book.

```
nltk.tokenize.sent_tokenize(text,  
                             language='english')    [source]
```

Return a sentence-tokenized copy of *text*, using NLTK's recommended sentence tokenizer (currently PunktSentenceTokenizer for the specified language).

Parameters

- **text** – text to split into sentences
- **language** – the model name in the Punkt corpus

```
nltk.tokenize.word_tokenize(text,  
                             language='english',  
                             preserve_line=False)    [source]
```

Return a tokenized copy of *text*, using NLTK's recommended word tokenizer (currently an improved TreebankWordTokenizer along with PunktSentenceTokenizer for the specified language).

Parameters

- **text** (*str*) – text to split into words
- **language** (*str*) – the model name in the Punkt corpus

- **preserve_line** (*bool*) – A flag to decide whether to sentence tokenize the text or not.

source 3.8.1 Jan 02,
// // 2023

© 2023, NLTK
Project

created with **Sphinx** and **NLTK**
Theme