


ХОЧУ ПОМОЧЬ ПРОЕКТУ

Модуль shlex в Python, анализ синтаксиса оболочки Unix



mango-office.ru

РЕКЛАМА

Виртуальная АТС Расширенная

2 000 ₽

Узнать больше

Статья

Анализ синтаксиса кавычек в кавычках (синтаксис оболочки Unix)

и для анализа синтаксиса оболочки Unix. Его можно использовать для написания кода или для анализа и разбора [строк](#) в кавычках.

Вводимым текстом является идентификация последовательности цитируемых слов как единого слова. Кавычки не всегда работает должным образом, особенно если есть вложенные уровни кавычек.

Наивным подходом было бы создание [регулярного выражения](#) для поиска частей текста вне кавычек, чтобы отделить их от текста внутри кавычек или наоборот. Это было бы излишне сложно и склонно к ошибкам, возникающим в результате крайних случаев, таких как апострофы или даже опечатки. Лучшее решение - использовать настоящий синтаксический анализатор, такой как предоставленный [модулем shlex](#).

## Правила разбора

При работе в режиме, отличном от POSIX, shlex будет пытаться соблюдать следующие правила:

- Символы кавычек не распознаются в словах Do"NotSeparate, разбирается как одно слово DoNotSeparate;
- [Escape-последовательности](#) не распознаются;
- Заключение символов в кавычки сохраняет буквальное значение всех символов внутри кавычек;
- Закрывающие кавычки отдельными словами "DoSeparate разделяется как "Do" и "Separate";
- Если sh.whitespace\_split имеет значение False, любой символ, не объявленный как символ слова, пробел или кавычка, будет возвращен как односимвольный токен. Если sh.whitespace\_split имеет значение True, то shlex будет разбивать слова только в пробелах;
- EOF сигнализируется пустой строкой '';
- Невозможно проанализировать пустые строки, даже если они заключены в кавычки.

При работе в режиме POSIX, shlex будет пытаться соблюдать следующие правила синтаксического анализа:

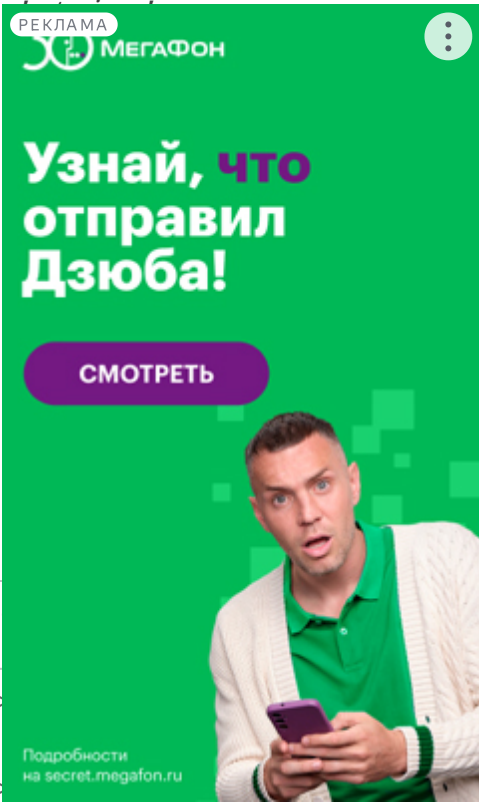
- Кавычки удаляются и не разделяют слова, например слово Do"NotSeparate будет выглядеть как DoNotSeparate
- Экранирующие символы без кавычек, например \ сохраняют буквальное значение символа, следующего за ним;
- Заключение символов в кавычки, которые не являются частью sh.escapedquotes, например '"', сохраняет буквальное значение всех символов внутри кавычек;
- Заключение в кавычки символов, являющихся частью sh.escapedquotes, например '"', сохраняет буквальное значение всех символов внутри кавычек, за исключением символов, упомянутых в shlex.escape. Экранирующие символы сохраняют свое особое значение только тогда, когда за ними следует используемая кавычка или сам экранирующий символ.
- EOF сигнализируется значением None;
- Пустые строки в кавычках '' разрешены.

## Примеры использования:

Разбор строки с кавычками:

```
import shlex
>>> text = """This string has embedded "double quotes" and
...         'single quotes' in it, and even "a 'nested example'"."""
>>> tokens = shlex.shlex(text)
```

```
>>> for token in lexer:
...     print('{!r}'.format(token))
...
# 'This'
```



Примеры использования escape-последовательностей и метасимволов в строках команд терминальной оболочки:

```
>>> command = f'ls -l {filename}'
>>> print(command)
# ls -l somefile; rm -rf ~

# quote() позволяет закрыть дыру в безопасности:
>>> command = f'ls -l {shlex.quote(filename)}'
>>> print(command)
# ls -l 'somefile; rm -rf ~'
```

Разбор строки с командой bash/sh на отдельные команды:

```
>>> import shlex, pprint
>>> cmd = '/bin/vikings -input eggs.txt -output "spam spam.txt" -cmd "echo \'$MONEY\'"'
>>> args = shlex.split(cmd)
>>> pprint.pprint(args, width='60')
#[ '/bin/vikings',
#   '-input',
#   'eggs.txt',
#   '-output',
#   'spam spam.txt',
#   '-cmd',
#   "echo '$MONEY'"]
```

### Содержание раздела:

- [КРАТКИЙ ОБЗОР МАТЕРИАЛА.](#)
- [Разбор команд bash, dash и sh модулем shlex](#)
- [Функции split\(\) модуля shlex](#)
- [Функция join\(\) модуля shlex](#)
- [Функция quote\(\) модуля shlex](#)
- [Класс shlex\(\) модуля shlex](#)

Вверх