urlextract

Release 1.8.0

Dec 19, 2022

Contents

urlextract is package with python class and command line script used for extraction of URLs from given text.

CHAPTER 1

urlextract - command line

urlextract - command line program that will print all URLs to stdout

Usage: `$ urlextract [-h] [-v] [-u] [-dl] [-c] [-i <ignore_file>] [-p <permit_file>] [-l LIMIT] [<input_file>]`

positional arguments: <input_file> input text file with URLs to extract

optional arguments:

-h, --help show this help message and exit

-v, --version show program's version number and exit

-u, --unique print out only unique URLs found in file

-dl, --disable-localhost disable extracting "localhost" as URL

-c, --check-dns print out only URLs for existing domain names

-i <ignore_file>, --ignore-file <ignore_file> input text file with URLs to exclude from extraction

-p <permit_file>, --permit-file <permit_file> input text file with URLs that can be processed

-l LIMIT, --limit LIMIT Maximum count of URLs that can be processed. Set 0 to disable the limit. Default: 10000

CHAPTER 2

URLExtract class

class urlextract.**URLExtract**(*extract_email=False*, *cache_dns=True*, *extract_localhost=True*, *limit=10000*, *allow_mixed_case_hostname=True*, ***kwargs*)

Class for finding and extracting URLs from given string.

Examples:

```
from urlextract import URLExtract

extractor = URLExtract()
urls = extractor.find_urls("Let's have URL example.com example.")
print(urls) # prints: ['example.com']

# Another way is to get a generator over found URLs in text:
for url in extractor.gen_urls(example_text):
    print(url) # prints: ['example.com']

# Or if you want to just check if there is at least one URL in text:
if extractor.has_urls(example_text):
    print("Given text contains some URL")
```

add_enclosure(*left_char: str*, *right_char: str*)

Add new enclosure pair of characters. That and should be removed when their presence is detected at beginning and end of found URL

Parameters

- **left_char** (*str*) – left character of enclosure pair - e.g. "("
- **right_char** (*str*) – right character of enclosure pair - e.g. ")"

allow_mixed_case_hostname

If set to True host should contain mixed case letters (upper-case and lower-case)

Return type bool

extract_email

If set to True email will be extracted from text

Return type bool

extract_localhost
If set to True 'localhost' will be extracted as URL from text

Return type bool

find_urls(*text: str*, *only_unique=False*, *check_dns=False*, *get_indices=False*, *with_schema_only=False*) → List[Union[str, Tuple[str, Tuple[int, int]]]]
Find all URLs in given text.

Parameters

- **text** (*str*) – text where we want to find URLs
- **only_unique** (*bool*) – return only unique URLs
- **check_dns** (*bool*) – filter results to valid domains
- **get_indices** (*bool*) – whether to return beginning and ending indices as (<url>, (idx_begin, idx_end))
- **with_schema_only** (*bool*) – get domains with schema only (e.g. https://janlipovsky.cz but not example.com)

Returns list of URLs found in text

Return type list

Raises **URLExtractError** – Raised when count of found URLs reaches given limit. Processed URLs are returned in *data* argument.

gen_urls(*text: str*, *check_dns=False*, *get_indices=False*, *with_schema_only=False*) → Generator[Union[str, Tuple[str, Tuple[int, int]]], None, None]
Creates generator over found URLs in given text.

Parameters

- **text** (*str*) – text where we want to find URLs
- **check_dns** (*bool*) – filter results to valid domains
- **get_indices** (*bool*) – whether to return beginning and ending indices as (<url>, (idx_begin, idx_end))
- **with_schema_only** (*bool*) – get domains with schema only

Yields URL or URL with indices found in text or empty string if nothing was found

Return type str|tuple(str, tuple(int, int))

get_after_tld_chars() → List[str]
Returns list of chars that are allowed after TLD

Returns list of chars that are allowed after TLD

Return type list

get_enclosures() → Set[Tuple[str, str]]
Returns set of enclosure pairs that might be used to enclosure URL. For example brackets (example.com), [example.com], {example.com}

Returns set of tuple of enclosure characters

Return type set(tuple(str,str))

get_stop_chars_left() → Set[str]
Returns set of stop chars for text on left from TLD.

Returns set of stop chars

Return type set

get_stop_chars_left_from_scheme() → Set[str]
Returns set of stop chars for text on left from scheme.

Returns set of stop chars

Return type set

get_stop_chars_right() → Set[str]
Returns set of stop chars for text on right from TLD.

Returns set of stop chars

Return type set

static get_version() → str
Returns version number.

Returns version number

Return type str

has_urls(*text: str*, *check_dns=False*, *with_schema_only=False*) → bool
Checks if text contains any valid URL. Returns True if text contains at least one URL.

Parameters

- **text** – text where we want to find URLs
- **check_dns** (*bool*) – filter results to valid domains
- **with_schema_only** (*bool*) – consider domains with schema only

Returns True if et least one URL was found, False otherwise

Return type bool

ignore_list
Set of URLs to be ignored (not returned) while extracting from text

Returns Returns set of ignored URLs

Return type set(str)

load_ignore_list(*file_name*)
Load URLs from file into ignore list

Parameters **file_name** (*str*) – path to file containing URLs

load_permit_list(*file_name*)
Load URLs from file into permit list

Parameters **file_name** (*str*) – path to file containing URLs

permit_list
Set of URLs that can be processed

Returns Returns set of URLs that can be processed

Return type set(str)

remove_enclosure(*left_char: str*, *right_char: str*)
Remove enclosure pair from set of enclosures.

Parameters

- **left_char** (*str*) – left character of enclosure pair - e.g. "("
- **right_char** (*str*) – right character of enclosure pair - e.g. ")"

set_after_tld_chars(*after_tld_chars: Iterable[str]*)
Set chars that are allowed after TLD.

Parameters **after_tld_chars** (*list*) – list of characters

set_stop_chars_left(*stop_chars: Set[str]*)
Set stop characters for text on left from TLD. Stop characters are used when determining end of URL.

Parameters **stop_chars** (*set*) – set of characters

Raises TypeError

set_stop_chars_left_from_scheme(*stop_chars: Set[str]*)
Set stop characters for text on left from scheme. Stop characters are used when determining end of URL.

Parameters **stop_chars** (*set*) – set of characters

Raises TypeError

set_stop_chars_right(*stop_chars: Set[str]*)
Set stop characters for text on right from TLD. Stop characters are used when determining end of URL.

Parameters **stop_chars** (*set*) – set of characters

Raises TypeError

update()
Update TLD list cache file.

Returns True if update was successful False otherwise

Return type bool

update_when_older(*days: int*) → bool
Update TLD list cache file if the list is older than number of days given in parameter *days* or if it does not exist.

Parameters **days** (*int*) – number of days from last change

Returns True if update was successful, False otherwise

Return type bool

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Index