Link Extractors

A link extractor is an object that extracts links from responses.

The `__init__` method of :class:`~scrapy.linkextractors.lxmlhtml.LxmlLinkExtractor` takes settings that determine which links may be extracted. :class:`LxmlLinkExtractor.extract_links <scrapy.linkextractors.lxmlhtml.LxmlLinkExtractor.extract_links>` returns a list of matching :class:`~scrapy.link.Link` objects from a :class:`~scrapy.http.Response` object.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst`, line 9); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst`, line 9); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst`, line 9); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst`, line 9); *backlink*

Unknown interpreted text role "class".

Link extractors are used in :class:`~scrapy.spiders.CrawlSpider` spiders through a set of :class:`~scrapy.spiders.Rule` objects.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst`, line 16); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst`, line 16); *backlink*

Unknown interpreted text role "class".

You can also use link extractors in regular spiders. For example, you can instantiate :class:`LinkExtractor <scrapy.linkextractors.lxmlhtml.LxmlLinkExtractor>` into a class variable in your spider, and use it from your spider callbacks:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst`, line 19); *backlink*

Unknown interpreted text role "class".

```
def parse(self, response):
    for link in self.link_extractor.extract_links(response):
        yield Request(link.url, callback=self.parse)
```

Link extractor reference

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst`, line 32)

Unknown directive type "module".

```
.. module:: scrapy.linkextractors
   :synopsis: Link extractors classes
```

The link extractor class is :class:`scrapy.linkextractors.lxmlhtml.LxmlLinkExtractor`. For convenience it can also be imported as `scrapy.linkextractors.LinkExtractor`:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst, line 35); *backlink*

Unknown interpreted text role "class".

```
from scrapy.linkextractors import LinkExtractor
```

LxmlLinkExtractor

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst, line 44)

Unknown directive type "module".

```
.. module:: scrapy.linkextractors.lxmlhtml
   :synopsis: lxml's HTMLParser-based link extractors
```

LxmlLinkExtractor is the recommended link extractor with handy filtering options. It is implemented using lxml's robust HTMLParser.

param allow: a single regular expression (or list of regular expressions) that the (absolute) urls must match in order to be extracted. If not given (or empty), it will match all links.

type allow: str or list

param deny: a single regular expression (or list of regular expressions) that the (absolute) urls must match in order to be excluded (i.e. not extracted). It has precedence over the `allow` parameter. If not given (or empty) it won't exclude any links.

type deny: str or list

param allow_domains: a single value or a list of string containing domains which will be considered for extracting the links

type allow_domains: str or list

param deny_domains: a single value or a list of strings containing domains which won't be considered for extracting the links

type deny_domains: str or list

param deny_extensions: a single value or list of strings containing extensions that should be ignored when extracting links. If not given, it will default to :data:`scrapy.linkextractors.IGNORED_EXTENSIONS`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst, line 72); *backlink*

Unknown interpreted text role "data".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst, line 77)

Unknown directive type "versionchanged".

```
.. versionchanged:: 2.0
   :data:`~scrapy.linkextractors.IGNORED_EXTENSIONS` now includes
   ``7z``, ``7zip``, ``apk``, ``bz2``, ``cdr``, ``dmg``, ``ico``,
   ``iso``, ``tar``, ``tar.gz``, ``webm``, and ``xz``.
```

type deny_extensions: list

param restrict_xpaths: is an XPath (or list of XPath's) which defines regions inside the response where links should be extracted from. If given, only the text selected by those XPath will be scanned for links. See examples below.

type restrict_xpaths: str or list

param restrict_css: a CSS selector (or list of selectors) which defines regions inside the response where links should be extracted from. Has the same behaviour as `restrict_xpaths`.

type restrict_css: str or list

param restrict_text:

a single regular expression (or list of regular expressions) that the link's text must match in order to be extracted. If not given (or empty), it will match all links. If a list of regular expressions is given, the link will be extracted if it matches at least one.

type restrict_text: str or list

param tags: a tag or a list of tags to consider when extracting links. Defaults to `('a', 'area')`.

type tags: str or list

param attrs: an attribute or list of attributes which should be considered when looking for links to extract (only for those tags specified in the `tags` parameter). Defaults to `('href',)`

type attrs: list

param canonicalize: canonicalize each extracted url (using w3lib.url.canonicalize_url). Defaults to `False`. Note that canonicalize_url is meant for duplicate checking; it can change the URL visible at server side, so the response can be different for requests with canonicalized and raw URLs. If you're using LinkExtractor to follow links it is more robust to keep the default `canonicalize=False`.

type canonicalize: bool

param unique: whether duplicate filtering should be applied to extracted links.

type unique: bool

param process_value: a function which receives each value extracted from the tag and attributes scanned and can modify the value and return a new one, or return `None` to ignore the link altogether. If not given, `process_value` defaults to `lambda x: x`.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst, line 127)**
>
> Unknown directive type "highlight".
>
> ```
> .. highlight:: html
> ```

For example, to extract links from this code:

```
<a href="javascript:goToPage('../other/page.html'); return false">Link text</a>
```

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst, line 133)**
>
> Unknown directive type "highlight".
>
> ```
> .. highlight:: python
> ```

You can use the following function in `process_value`:

```
def process_value(value):
    m = re.search("javascript:goToPage\('(.*?)'", value)
    if m:
        return m.group(1)
```

type process_value: collections.abc.Callable

param strip: whether to strip whitespaces from extracted attributes. According to HTML5 standard, leading and trailing whitespaces must be stripped from `href` attributes of `<a>`, `<area>` and many other elements, `src` attribute of `<img>`, `<iframe>` elements, etc., so LinkExtractor strips space chars by default. Set `strip=False` to turn it off (e.g. if you're extracting urls from elements or attributes which allow leading/trailing whitespaces).

type strip: bool

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst, line 153)**
>
> Unknown directive type "automethod".
>
> ```
> .. automethod:: extract_links
> ```

Link

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst, line 158)

Unknown directive type "module".

```
.. module:: scrapy.link
   :synopsis: Link from link extractors
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]link-extractors.rst, line 161)

Unknown directive type "autoclass".

```
.. autoclass:: Link
```