Selecting dynamically-loaded content

Some webpages show the desired data when you load them in a web browser. However, when you download them using Scrapy, you cannot reach the desired data using :ref:`selectors <topics-selectors>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 7); ***backlink***

Unknown interpreted text role "ref".

When this happens, the recommended approach is to :ref:`find the data source <topics-finding-data-source>` and extract the data from it.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 11); ***backlink***

Unknown interpreted text role "ref".

If you fail to do that, and you can nonetheless access the desired data through the :ref:`DOM <topics-livedom>` from your web browser, see :ref:`topics-javascript-rendering`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 15); ***backlink***

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 15); ***backlink***

Unknown interpreted text role "ref".

Finding the data source

To extract the desired data, you must first find its source location.

If the data is in a non-text-based format, such as an image or a PDF document, use the :ref:`network tool <topics-network-tool>` of your web browser to find the corresponding request, and :ref:`reproduce it <topics-reproducing-requests>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 26); ***backlink***

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 26); ***backlink***

Unknown interpreted text role "ref".

If your web browser lets you select the desired data as text, the data may be defined in embedded JavaScript code, or loaded from an external resource in a text-based format.

In that case, you can use a tool like wgrep to find the URL of that resource.

If the data turns out to come from the original URL itself, you must :ref:`inspect the source code of the webpage <topics-inspecting-source>` to determine where the data is located.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 37); ***backlink***

Unknown interpreted text role "ref".

If the data comes from a different URL, you will need to :ref:`reproduce the corresponding request <topics-reproducing-requests>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 41); ***backlink***

Unknown interpreted text role "ref".

Inspecting the source code of a webpage

Sometimes you need to inspect the source code of a webpage (not the :ref:`DOM <topics-livedom>`) to determine where some desired data is located.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 49**); ***backlink***

Unknown interpreted text role "ref".

Use Scrapyâ€™s :command:`fetch` command to download the webpage contents as seen by Scrapy:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 52**); ***backlink***

Unknown interpreted text role "command".

```
scrapy fetch --nolog https://example.com > response.html
```

If the desired data is in embedded JavaScript code within a `<script/>` element, see :ref:`topics-parsing-javascript`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 57**); ***backlink***

Unknown interpreted text role "ref".

If you cannot find the desired data, first make sure itâ€™s not just Scrapy: download the webpage with an HTTP client like curl or wget and see if the information can be found in the response they get.

If they get a response with the desired data, modify your Scrapy :class:`~scrapy.Request` to match that of the other HTTP client. For example, try using the same user-agent string (:setting:`USER_AGENT`) or the same :attr:`~scrapy.Request.headers`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 64**); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 64**); ***backlink***

Unknown interpreted text role "setting".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 64**); ***backlink***

Unknown interpreted text role "attr".

If they also get a response without the desired data, youâ€™ll need to take steps to make your request more similar to that of the web browser. See :ref:`topics-reproducing-requests`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 69**); ***backlink***

Unknown interpreted text role "ref".

Reproducing requests

Sometimes we need to reproduce a request the way our web browser performs it.

Use the :ref:`network tool <topics-network-tool>` of your web browser to see how your web browser performs the desired request, and try to reproduce that request with Scrapy.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 80**); ***backlink***

Unknown interpreted text role "ref".

It might be enough to yield a :class:`~scrapy.Request` with the same HTTP method and URL. However, you may also need to reproduce the body, headers and form parameters (see :class:`~scrapy.FormRequest`) of that request.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 84**); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 84**); ***backlink***

Unknown interpreted text role "class".

As all major browsers allow to export the requests in cURL format, Scrapy incorporates the method :meth:`~scrapy.Request.from_curl()` to generate an equivalent :class:`~scrapy.Request` from a cURL command. To get more information visit :ref:`request from curl <requests-from-curl>` inside the network tool section.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 88**); ***backlink***

Unknown interpreted text role "meth".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 88**); ***backlink***

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 88**); ***backlink***

Unknown interpreted text role "ref".

Once you get the expected response, you can :ref:`extract the desired data from it <topics-handling-response-formats>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 95**); ***backlink***

Unknown interpreted text role "ref".

You can reproduce any request with Scrapy. However, some times reproducing all necessary requests may not seem efficient in developer time. If that is your case, and crawling speed is not a major concern for you, you can alternatively consider :ref:`JavaScript pre-rendering <topics-javascript-rendering>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 98**); ***backlink***

Unknown interpreted text role "ref".

If you get the expected response *sometimes*, but not always, the issue is probably not your request, but the target server. The target server might be buggy, overloaded, or :ref:`banning <bans>` some of your requests.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, **line 103**); ***backlink***

Unknown interpreted text role "ref".

Note that to translate a cURL command into a Scrapy request, you may use curl2scrapy.

Handling different response formats

Once you have a response with the desired data, how you extract the desired data from it depends on the type of response:

- If the response is HTML or XML, use :ref:`selectors <topics-selectors>` as usual.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 118); *backlink*

Unknown interpreted text role "ref".

- If the response is JSON, use :func:`json.loads` to load the desired data from :attr:`response.text <scrapy.http.TextResponse.text>`:

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 121); *backlink*

  Unknown interpreted text role "func".

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 121); *backlink*

  Unknown interpreted text role "attr".

  ```
  data = json.loads(response.text)
  ```

  If the desired data is inside HTML or XML code embedded within JSON data, you can load that HTML or XML code into a :class:`~scrapy.Selector` and then :ref:`use it <topics-selectors>` as usual:

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 126); *backlink*

  Unknown interpreted text role "class".

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 126); *backlink*

  Unknown interpreted text role "ref".

  ```
  selector = Selector(data['html'])
  ```

- If the response is JavaScript, or HTML with a `<script/>` element containing the desired data, see :ref:`topics-parsing-javascript`.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 133); *backlink*

  Unknown interpreted text role "ref".

- If the response is CSS, use a :doc:`regular expression <library/re>` to extract the desired data from :attr:`response.text <scrapy.http.TextResponse.text>`.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 136); *backlink*

  Unknown interpreted text role "doc".

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 136); *backlink*

  Unknown interpreted text role "attr".

- If the response is an image or another format based on images (e.g. PDF), read the response as bytes from :attr:`response.body <scrapy.http.TextResponse.body>` and use an OCR solution to extract the desired data as text.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 142); ***backlink***

Unknown interpreted text role "attr".

For example, you can use pytesseract. To read a table from a PDF, tabula-py may be a better choice.

- If the response is SVG, or HTML with embedded SVG containing the desired data, you may be able to extract the desired data using :ref:`selectors <topics-selectors>`, since SVG is based on XML.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 150); ***backlink***

Unknown interpreted text role "ref".

Otherwise, you might need to convert the SVG code into a raster image, and :ref:`handle that raster image <topics-parsing-images>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 154); ***backlink***

Unknown interpreted text role "ref".

Parsing JavaScript code

If the desired data is hardcoded in JavaScript, you first need to get the JavaScript code:

- If the JavaScript code is in a JavaScript file, simply read :attr:`response.text <scrapy.http.TextResponse.text>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 165); ***backlink***

Unknown interpreted text role "attr".

- If the JavaScript code is within a `<script/>` element of an HTML page, use :ref:`selectors <topics-selectors>` to extract the text within that `<script/>` element.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 168); ***backlink***

Unknown interpreted text role "ref".

Once you have a string with the JavaScript code, you can extract the desired data from it:

- You might be able to use a :doc:`regular expression <library/re>` to extract the desired data in JSON format, which you can then parse with :func:`json.loads`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 175); ***backlink***

Unknown interpreted text role "doc".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 175); ***backlink***

Unknown interpreted text role "func".

For example, if the JavaScript code contains a separate line like `var data = {"field": "value"};` you can extract that data as follows:

```
>>> pattern = r'\bvar\s+data\s*=\s*(\{.*?\})\s*;\s*\n'
>>> json_data = response.css('script::text').re_first(pattern)
>>> json.loads(json_data)
{'field': 'value'}
```

- chompjs provides an API to parse JavaScript objects into a :class:`dict`.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 187); *backlink*

  Unknown interpreted text role "class".

  For example, if the JavaScript code contains `var data = {field: "value", secondField: "second value"};` you can extract that data as follows:

  ```
  >>> import chompjs
  >>> javascript = response.css('script::text').get()
  >>> data = chompjs.parse_js_object(javascript)
  >>> data
  {'field': 'value', 'secondField': 'second value'}
  ```

- Otherwise, use js2xml to convert the JavaScript code into an XML document that you can parse using :ref:`selectors <topics-selectors>`.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 199); *backlink*

  Unknown interpreted text role "ref".

  For example, if the JavaScript code contains `var data = {field: "value"};` you can extract that data as follows:

  ```
  >>> import js2xml
  >>> import lxml.etree
  >>> from parsel import Selector
  >>> javascript = response.css('script::text').get()
  >>> xml = lxml.etree.tostring(js2xml.parse(javascript), encoding='unicode')
  >>> selector = Selector(text=xml)
  >>> selector.css('var[name="data"]').get()
  '<var name="data"><object><property name="field"><string>value</string></property></object></var>'
  ```

Pre-rendering JavaScript

On webpages that fetch data from additional requests, reproducing those requests that contain the desired data is the preferred approach. The effort is often worth the result: structured, complete data with minimum parsing time and network transfer.

However, sometimes it can be really hard to reproduce certain requests. Or you may need something that no request can give you, such as a screenshot of a webpage as seen in a web browser.

In these cases use the Splash JavaScript-rendering service, along with scrapy-splash for seamless integration.

Splash returns as HTML the :ref:`DOM <topics-livedom>` of a webpage, so that you can parse it with :ref:`selectors <topics-selectors>`. It provides great flexibility through configuration or scripting.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 231); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 231); *backlink*

Unknown interpreted text role "ref".

If you need something beyond what Splash offers, such as interacting with the DOM on-the-fly from Python code instead of using a previously-written script, or handling multiple web browser windows, you might need to :ref:`use a headless browser <topics-headless-browsing>` instead.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 235); *backlink*

Unknown interpreted text role "ref".

Using a headless browser

A headless browser is a special web browser that provides an API for automation. By installing the :ref:`asyncio reactor <install-asyncio>`, it is possible to integrate `asyncio`-based libraries which handle headless browsers.

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\scrapy-master\docs\topics\[scrapy-master][docs][topics]dynamic-content.rst`, line 248); *backlink***
>
> Unknown interpreted text role "ref".

One such library is playwright-python (an official Python port of playwright). The following is a simple snippet to illustrate its usage within a Scrapy spider:

```
import scrapy
from playwright.async_api import async_playwright

class PlaywrightSpider(scrapy.Spider):
    name = "playwright"
    start_urls = ["data:,"]  # avoid using the default Scrapy downloader

    async def parse(self, response):
        async with async_playwright() as pw:
            browser = await pw.chromium.launch()
            page = await browser.new_page()
            await page.goto("https:/example.org")
            title = await page.title()
            return {"title": title}
```

However, using playwright-python directly as in the above example circumvents most of the Scrapy components (middlewares, dupefilter, etc). We recommend using scrapy-playwright for a better integration.