
zyte-common-items

Release 0.19.0

unknown

Apr 24, 2024

GETTING STARTED

1	Setup	3
2	Items	5
3	Page objects	9
4	Field processors	11
5	Request templates	13
6	Reference	17
7	Changelog	57
8	Contributing	63
	Python Module Index	65
	Index	67

zyte-common-items is a Python 3.8+ library of [item](#) and [page object](#) classes for web data extraction that we use at [Zyte](#) to maximize opportunities for code reuse.

1.1 Installation

```
pip install zyte-common-items
```

1.2 Configuration

To allow `itemadapter` users, like `Scrapy`, to interact with *items*, prepend `ZyteItemAdapter` or `ZyteItemKeepEmptyAdapter` to `itemadapter.ItemAdapter.ADAPTER_CLASSES` as early as possible in your code:

```
from itemadapter import ItemAdapter
from zyte_common_items import ZyteItemAdapter

ItemAdapter.ADAPTER_CLASSES.appendleft(ZyteItemAdapter)
```

Alternatively, make your own subclass of `itemadapter.ItemAdapter`:

```
from collections import deque

from itemadapter import ItemAdapter
from zyte_common_items import ZyteItemAdapter

class MyItemAdapter(ItemAdapter):
    ADAPTER_CLASSES = deque([ZyteItemAdapter]) + ItemAdapter.ADAPTER_CLASSES
```

Now you can use `MyItemAdapter` where you would use `itemadapter.ItemAdapter`.

The *provided item classes* can be used to map data extracted from web pages, e.g. using *page objects*.

2.1 Creating items from dictionaries

You can create an *item* from any *dict*-like object via the *from_dict()* method.

For example, to create a *Product*:

```
>>> from zyte_common_items import Product
>>> data = {
...     'url': 'https://example.com/',
...     'mainImage': {
...         'url': 'https://example.com/image.png',
...     },
...     'gtin': [
...         {'type': 'gtin13', 'value': '9504000059446'},
...     ],
... }
>>> product = Product.from_dict(data)
```

from_dict() applies the right classes to nested data, such as *Image* and *Gtin* for the input above.

```
>>> product.url
'https://example.com/'
>>> product.mainImage
Image(url='https://example.com/image.png')
>>> product.canonicalUrl
>>> product.gtin
[Gtin(type='gtin13', value='9504000059446')]
```

2.2 Creating items from lists

You can create items in bulk using the `from_list()` method:

```
>>> from zyte_common_items import Product
>>> data_list = [
...     {'url': 'https://example.com/1', 'name': 'Product 1'},
...     {'url': 'https://example.com/2', 'name': 'Product 2'},
...     {'url': 'https://example.com/3', 'name': 'Product 3'},
...     {'url': 'https://example.com/4', 'name': 'Product 4'}
... ]
>>> products = Product.from_list(data_list)
>>> len(products)
4
>>> products[0].url
'https://example.com/1'
>>> products[3].name
'Product 4'
```

This can be especially useful if you're processing lots of items from an API, file, database, etc.

2.3 Handling unknown fields

Items and *components* do not allow attributes beyond those they define:

```
>>> from zyte_common_items import Product
>>> product = Product(url="https://example.com", foo="bar")
Traceback (most recent call last):
...
TypeError: ... got an unexpected keyword argument 'foo'
>>> product = Product(url="https://example.com")
>>> product.foo = "bar"
Traceback (most recent call last):
...
AttributeError: 'Product' object has no attribute 'foo'
```

However, when using `from_dict()` and `from_list()`, unknown fields assigned to items and components **won't cause an error**. Instead, they are placed inside the `_unknown_fields_dict` attribute, and can be accessed the same way as known fields using `ZyteItemAdapter`:

```
>>> from zyte_common_items import Product, ZyteItemAdapter
>>> data = {
...     'url': 'https://example.com/',
...     'unknown_field': True,
... }
>>> product = Product.from_dict(data)
>>> product._unknown_fields_dict
{'unknown_field': True}
>>> adapter = ZyteItemAdapter(product)
>>> adapter['unknown_field']
True
```

This allows compatibility with future field changes in the input data, which could cause backwards incompatibility issues.

Note, however, that unknown fields are only supported within items and components. Input processing can still fail for other types of unexpected input:

```
>>> from zyte_common_items import Product
>>> data = {
...     'url': 'https://example.com/',
...     'mainImage': 'not a dictionary',
... }
>>> product = Product.from_dict(data)
Traceback (most recent call last):
...
ValueError: Expected mainImage to be a dict with fields from zyte_common_items.
↳ components.media.Image, got 'not a dictionary'.
>>> data = {
...     'url': 'https://example.com/',
...     'breadcrumbs': 3,
... }
>>> product = Product.from_dict(data)
Traceback (most recent call last):
...
ValueError: Expected breadcrumbs to be a list, got 3.
```

2.4 Defining custom items

You can subclass *Item* or any *item subclass* to define your own item.

Item is a *slotted attrs class* and, to enjoy the benefits of that, subclasses should also be *slotted attrs classes*. For example:

```
>>> import attrs
>>> from zyte_common_items import Item
>>> @attrs.define
... class CustomItem(Item):
...     foo: str
```

Mind that *slotted attrs classes* do not support multiple inheritance.

PAGE OBJECTS

Built-in page object classes are good base classes for custom page object classes that implement website-specific *page* objects.

They provide the following base line:

- They declare the *item class* that they return, allowing for their `to_item` method to automatically build an instance of it from `@field`-decorated methods. See *Fields*.
- They provide a default implementation for their *metadata* and *url* fields.
- They also provide a default implementation for some item-specific fields in pages that have those (except for description in the pages for *Article* which has different requirements):
 - *currency*
 - *currencyRaw*
 - *description*
 - *descriptionHtml*

The following code shows a *ProductPage* subclass whose `to_item` method returns an instance of *Product* with *metadata*, a *name*, and a *url*:

```
import attrs
from zyte_common_items import ProductPage

class CustomProductPage(ProductPage):
    @field
    def name(self):
        return self.css("h1::text").get()
```

Page object classes with the `Auto` prefix can be used to easily define page object classes that get an *item* as a dependency from another page object class, can generate an identical item by default, and can also easily override specific fields of the item, or even return a new item with extra fields. For example:

```
import attrs
from web_poet import Returns, field
from zyte_common_items import AutoProductPage, Product

@attrs.define
class ExtendedProduct(Product):
    foo: str
```

(continues on next page)

(continued from previous page)

```
class ExtendedProductPage(AutoProductPage, Returns[ExtendedProduct]):
    @field
    def name(self):
        return f"{self.product.brand.name} {self.product.name}"

    @field
    def foo(self):
        return "bar"
```

3.1 Extractors

For some nested fields (*ProductFromList*, *ProductVariant*), *base extractors* exist that you can subclass to write your own extractors.

They provide the following base line:

- They declare the *item class* that they return, allowing for their `to_item` method to automatically build an instance of it from `@field`-decorated methods. See *Fields*.
- They also provide default *processors* for some item-specific fields.

See *Extractor API*.

FIELD PROCESSORS

4.1 Overview

This library provides useful *field processors* ([web-poet documentation](#)) and complementary *mixins*. Built-in *page object classes* and *extractor classes* use them by default for the *corresponding fields*.

By design, the processors enabled by default are “transparent”: they don’t change the output of the field if the result is of the expected final type. For example, if there is a `str` attribute in the item, and the field returns `str` value, the default processor returns the value as-is.

Usually, to engage a *built-in field processor*, a field must return a `Selector`, `SelectorList`, or `HtmlElement` object. Then the field processor takes care of extracting the right data.

4.2 Field mapping

The following table indicates which fields use which processors by default in *built-in page object classes* and *extractor classes*:

Field	Default processor
<code>aggregateRating</code>	<code>rating_processor()</code>
<code>brand</code>	<code>brand_processor()</code>
<code>breadcrumbs</code>	<code>breadcrumbs_processor()</code>
<code>description</code> (excluding articles)	<code>description_processor()</code>
<code>descriptionHtml</code>	<code>description_html_processor()</code>
<code>gtin</code>	<code>gtin_processor()</code>
<code>price</code>	<code>price_processor()</code>
<code>regularPrice</code>	<code>simple_price_processor()</code>

4.3 Examples

Here are examples of inputs and matching field implementations that work on built-in page object and extractor classes:

Input HTML fragment	Field implementation and output
<pre> 3.8 (7 reviews) </pre>	<pre>@field def aggregateRating(self): return self.css(".reviews")</pre> <pre>Product(aggregateRating=AggregateRating(bestRating=None, ratingValue=3.8, reviewCount=7,),)</pre> <p>Supports separate selectors per field. See rating_processor().</p>
<pre><p class="brand"> </p></pre>	<pre>@field def brand(self): return self.css(".brand")</pre> <pre>Product(brand="Some Brand",)</pre>
<pre><div class="nav"> Home About </div></pre>	<pre>@field def breadcrumbs(self): return self.css(".nav")</pre> <pre>Product(breadcrumbs=[Breadcrumb(name="Home", url="https://example.com/home ↪",), Breadcrumb(name="About", url="https://example.com/about ↪",),],)</pre>
<pre><div class="desc"> <p>Ideal for scraping glass.</p> <p>Durable and reusable.</p> </div></pre>	<pre>@field def descriptionHtml(self): return self.css(".desc")</pre> <pre>Product(description=("Ideal for scraping glass.\n" "\n" "Durable and reusable."), descriptionHtml=("<article>\n" </pre>
12	Chapter 4. Field processors

REQUEST TEMPLATES

Request templates are *items* that allow writing reusable code that creates *Request* objects from parameters.

5.1 Using request templates

After you *write a request template page object* for a website, you can get a request template item for that website and call its `request` method to build a request with specific parameters. For example:

```
from scrapy import Request, Spider
from scrapy_poet import DummyResponse
from zyte_common_items import SearchRequestTemplate

class ExampleComSpider(Spider):
    name = "example_com"

    def start_requests(self):
        yield Request("https://example.com", callback=self.start_search)

    def start_search(
        self, response: DummyResponse, search_request_template: SearchRequestTemplate
    ):
        yield search_request_template.request(keyword="foo bar").to_scrapy(
            callback=self.parse_result
        )

    def parse_result(self, response): ...
```

`search_request_template.request(keyword="foo bar")` builds a *Request* object, e.g. with URL `https://example.com/search?q=foo+bar`.

5.2 Writing a request template page object

To enable building a request template for a given website, build a page object for that website that returns the corresponding request template item class. For example:

```
from web_poet import handle_urls
from zyte_common_items import SearchRequestTemplatePage

@handle_urls("example.com")
class ExampleComSearchRequestTemplatePage(SearchRequestTemplatePage):
    @field
    def url(self):
        return "https://example.com/search?q={{ keyword|quote_plus }}"
```

Strings returned by request template page object fields are [Jinja templates](#), and may use the keyword arguments of the `request` method of the corresponding *request template item class*.

Often, you only need to build a URL template by figuring out where request parameters go and using the right URL-encoding filter, `urlencode()` or `quote_plus()`, depending on how spaces are encoded:

Example search URL for “foo bar”	URL template
<code>https://example.com/?q=foo%20bar</code>	<code>https://example.com/?q={{ keyword urlencode }}</code>
<code>https://example.com/?q=foo+bar</code>	<code>https://example.com/?q={{ keyword quote_plus }}</code>

You can use any of Jinja’s built-in filters, plus `quote_plus()`, and all other Jinja features. Jinja enables very complex scenarios:

```
class ComplexSearchRequestTemplatePage(SearchRequestTemplatePage):
    @field
    def url(self):
        return """
        {%-
            if keyword|length > 1
            and keyword[0]|lower == 'p'
            and keyword[1:]|int(-1) != -1
        -%}
        https://example.com/p/{{ keyword|upper }}
        {%- else -%}
        https://example.com/search
        {%- endif -%}
        """

    @field
    def method(self):
        return """
        {%-
            if keyword|length > 1
            and keyword[0]|lower == 'p'
            and keyword[1:]|int(-1) != -1
        -%}
        GET
        {%- else -%}
```

(continues on next page)

(continued from previous page)

```

        POST
        {%- endif -%}
    """

    @field
    def body(self):
        return """
            {%-
                if keyword|length > 1
                and keyword[0]|lower == 'p'
                and keyword[1:]|int(-1) != -1
            -%}
            {%- else -%}
                {"query": {{ keyword|tojson }}}
            {%- endif -%}
        """

    @field
    def headers(self):
        return [
            Header(
                name=(
                    """
                        {%-
                            if keyword|length > 1
                            and keyword[0]|lower == 'p'
                            and keyword[1:]|int(-1) != -1
                        -%}
                        {%- else -%}
                            Query
                        {%- endif -%}
                    """
                ),
                value="{{ keyword }}",
            ),
        ]

```


REFERENCE

6.1 Item API

6.1.1 Product

class `zyte_common_items.Product(**kwargs)`

Product from an e-commerce website.

`url` is the only required attribute.

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → *Optional*[float]

Returns the item probability if available, otherwise None.

additionalProperties: *Optional*[List[*AdditionalProperty*]]

List of name-value pairs of data about a specific, otherwise unmapped feature.

Additional properties usually appear in product pages in the form of a specification table or a free-form specification list.

Additional properties that require 1 or more extra requests may not be extracted.

See also `features`.

aggregateRating: *Optional*[*AggregateRating*]

Aggregate data about reviews and ratings.

availability: *Optional*[str]

Availability status.

The value is expected to be one of: "InStock", "OutOfStock".

brand: *Optional*[*Brand*]

Brand.

breadcrumbs: *Optional*[List[*Breadcrumb*]]

Webpage `breadcrumb` trail.

canonicalUrl: `Optional[str]`

Canonical form of the URL, as indicated by the website.

See also `url`.

color: `Optional[str]`

Color.

It is extracted as displayed (e.g. "white").

See also `size`, `style`.

currency: `Optional[str]`

Price currency [ISO 4217](#) alphabetic code (e.g. "USD").

See also `currencyRaw`.

currencyRaw: `Optional[str]`

Price currency as it appears on the webpage (no post-processing), e.g. "\$".

See also `currency`.

description: `Optional[str]`

Plain-text description.

If the description is split across different parts of the source webpage, only the main part, containing the most useful pieces of information, should be extracted into this attribute.

It may contain data found in other attributes (`features`, `additionalProperties`).

Format-wise:

- Line breaks and non-ASCII characters are allowed.
- There is no length limit for this attribute, the content should not be truncated.
- There should be no whitespace at the beginning or end.

See also `descriptionHtml`.

descriptionHtml: `Optional[str]`

HTML description.

See `description` for extraction details.

The format is not the raw HTML from the source webpage. See the [HTML normalization specification](#) for details.

features: `Optional[List[str]]`

List of features.

They are usually listed as bullet points in product webpages.

See also `additionalProperties`.

gtin: `Optional[List[Gtin]]`

List of standardized [GTIN](#) product identifiers associated with the product, which are unique for the product across different sellers.

See also: `mpn`, `productId`, `sku`.

images: `Optional[List[Image]]`

All product images.

The main image (see `mainImage`) should be first in the list.

Images only displayed as part of the product description are excluded.

mainImage: `Optional[Image]`

Main product image.

metadata: `Optional[ProductMetadata]`

Data extraction process metadata.

mpn: `Optional[str]`

Manufacturer part number (MPN).

A product should have the same MPN across different e-commerce websites.

See also: `gtin`, `productId`, `sku`.

name: `Optional[str]`

Name as it appears on the webpage (no post-processing).

price: `Optional[str]`

Price at which the product is being offered.

It is a string with the price amount, with a full stop as decimal separator, and no thousands separator or currency (see `currency` and `currencyRaw`), e.g. "10500.99".

If `regularPrice` is not `None`, `price` should always be lower than `regularPrice`.

productId: `Optional[str]`

Product identifier, unique within an e-commerce website.

It may come in the form of an SKU or any other identifier, a hash, or even a URL.

See also: `gtin`, `mpn`, `sku`.

regularPrice: `Optional[str]`

Price at which the product was being offered in the past, and which is presented as a reference next to the current price.

It may be labeled as the original price, the list price, or the maximum retail price for which the product is sold.

See `price` for format details.

If `regularPrice` is not `None`, it should always be higher than `price`.

size: `Optional[str]`

Size or dimensions.

Pertinent to products such as garments, shoes, accessories, etc.

It is extracted as displayed (e.g. "XL").

See also `color`, `style`.

sku: `Optional[str]`

Stock keeping unit (SKU) identifier, i.e. a merchant-specific product identifier.

See also: `gtin`, `mpn`, `productId`.

style: `Optional[str]`

Style.

Pertinent to products such as garments, shoes, accessories, etc.

It is extracted as displayed (e.g. "polka dots").

See also `color`, `size`.

url: `str`

Main URL from which the data has been extracted.

See also `canonicalUrl`.

variants: `Optional[List[ProductVariant]]`

List of variants.

When slightly different versions of a product are displayed on the same product page, allowing you to choose a specific product version from a selection, each of those product versions are considered a product variant.

Product variants usually differ in `color` or `size`.

The following items are *not* considered product variants:

- Different products within the same bundle of products.
- Product add-ons, e.g. premium upgrades of a base product.

Only variant-specific data is extracted as product variant details. For example, if variant-specific versions of the product description do not exist in the source webpage, the description attributes of the product variant are *not* filled with the base product description.

Extracted product variants may not include those that are not visible in the source webpage.

Product variant details may not include those that require multiple additional requests (e.g. 1 or more requests per variant).

class `zyte_common_items.ProductVariant(**kwargs)`

Product variant.

See *Product.variants*, *ProductVariantExtractor*, *ProductVariantSelectorExtractor*.

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → `Optional[float]`

Returns the item probability if available, otherwise `None`.

additionalProperties: `Optional[List[AdditionalProperty]]`

List of name-value pairs of data about a specific, otherwise unmapped feature.

Additional properties usually appear in product pages in the form of a specification table or a free-form specification list.

Additional properties that require 1 or more extra requests may not be extracted.

See also `features`.

availability: `Optional[str]`

Availability status.

The value is expected to be one of: "InStock", "OutOfStock".

canonicalUrl: `Optional[str]`

Canonical form of the URL, as indicated by the website.

See also `url`.

color: `Optional[str]`

Color.

It is extracted as displayed (e.g. "white").

See also `size`, `style`.

currency: `Optional[str]`

Price currency [ISO 4217](#) alphabetic code (e.g. "USD").

See also `currencyRaw`.

currencyRaw: `Optional[str]`

Price currency as it appears on the webpage (no post-processing), e.g. "\$".

See also `currency`.

gtin: `Optional[List[Gtin]]`

List of standardized [GTIN](#) product identifiers associated with the product, which are unique for the product across different sellers.

See also: `mpn`, `productId`, `sku`.

images: `Optional[List[Image]]`

All product images.

The main image (see `mainImage`) should be first in the list.

Images only displayed as part of the product description are excluded.

mainImage: `Optional[Image]`

Main product image.

mpn: `Optional[str]`

[Manufacturer part number](#) (MPN).

A product should have the same MPN across different e-commerce websites.

See also: `gtin`, `productId`, `sku`.

name: `Optional[str]`

Name as it appears on the webpage (no post-processing).

price: `Optional[str]`

Price at which the product is being offered.

It is a string with the price amount, with a full stop as decimal separator, and no thousands separator or currency (see `currency` and `currencyRaw`), e.g. "10500.99".

If `regularPrice` is not `None`, `price` should always be lower than `regularPrice`.

productId: `Optional[str]`

Product identifier, unique within an e-commerce website.

It may come in the form of an SKU or any other identifier, a hash, or even a URL.

See also: `gtin`, `mpn`, `sku`.

regularPrice: `Optional[str]`

Price at which the product was being offered in the past, and which is presented as a reference next to the current price.

It may be labeled as the original price, the list price, or the maximum retail price for which the product is sold.

See `price` for format details.

If `regularPrice` is not `None`, it should always be higher than `price`.

size: `Optional[str]`

Size or dimensions.

Pertinent to products such as garments, shoes, accessories, etc.

It is extracted as displayed (e.g. "XL").

See also `color`, `style`.

sku: `Optional[str]`

Stock keeping unit (SKU) identifier, i.e. a merchant-specific product identifier.

See also: `gtin`, `mpn`, `productId`.

style: `Optional[str]`

Style.

Pertinent to products such as garments, shoes, accessories, etc.

It is extracted as displayed (e.g. "polka dots").

See also `color`, `size`.

url: `Optional[str]`

Main URL from which the product variant data could be extracted.

See also `canonicalUrl`.

class `zyte_common_items.ProductMetadata(**kwargs)`

Metadata class for `zyte_common_items.Product.metadata`.

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: `YYYY-MM-DDThh:mm:ssZ`.

probability: `Optional[float]`

The probability (0 for 0%, 1 for 100%) that the resource features the expected data type.

For example, if the extraction of a product from a given URL is requested, and that URL points to the webpage of a product with complete certainty, the value should be `1`. If with complete certainty the webpage features a job listing instead of a product, the value should be `0`. When there is no complete certainty, the value could be anything in between (e.g. `0.96`).

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

6.1.2 Product list

class `zyte_common_items.ProductList(**kwargs)`

Product list from a product listing page of an e-commerce webpage.

It represents, for example, a single page from a category.

`url` is the only required attribute.

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → *Optional*[float]

Returns the item probability if available, otherwise None.

breadcrumbs: *Optional*[List[Breadcrumb]]

Webpage breadcrumb trail.

canonicalUrl: *Optional*[str]

Canonical form of the URL, as indicated by the website.

See also `url`.

categoryName: *Optional*[str]

Name of the product listing as it appears on the webpage (no post-processing).

For example, if the webpage is one of the pages of the Robots category, `categoryName` is 'Robots'.

metadata: *Optional*[ProductListMetadata]

Data extraction process metadata.

pageNumber: *Optional*[int]

Current page number, if displayed explicitly on the list page.

Numeration starts with 1.

paginationNext: *Optional*[Link]

Link to the next page.

products: *Optional*[List[ProductFromList]]

List of products.

It only includes product information found in the product listing page itself. Product information that requires visiting each product URL is not meant to be covered.

The order of the products reflects their position on the rendered page. Product order is top-to-bottom, and left-to-right or right-to-left depending on the webpage locale.

url: str

Main URL from which the data has been extracted.

See also `canonicalUrl`.

class `zyte_common_items.ProductFromList(**kwargs)`

Product from a product list from a product listing page of an e-commerce webpage.

See *ProductList*, *ProductFromListExtractor*, *ProductFromListSelectorExtractor*.

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → *Optional*[float]

Returns the item probability if available, otherwise None.

currency: *Optional*[str]

Price currency ISO 4217 alphabetic code (e.g. "USD").

See also `currencyRaw`.

currencyRaw: *Optional*[str]

Price currency as it appears on the webpage (no post-processing), e.g. "\$".

See also `currency`.

mainImage: *Optional*[Image]

Main product image.

metadata: *Optional*[ProbabilityMetadata]

Data extraction process metadata.

name: *Optional*[str]

Name as it appears on the webpage (no post-processing).

price: *Optional*[str]

Price at which the product is being offered.

It is a string with the price amount, with a full stop as decimal separator, and no thousands separator or currency (see `currency` and `currencyRaw`), e.g. "10500.99".

If `regularPrice` is not None, price should always be lower than `regularPrice`.

productId: *Optional*[str]

Product identifier, unique within an e-commerce website.

It may come in the form of an SKU or any other identifier, a hash, or even a URL.

regularPrice: *Optional*[str]

Price at which the product was being offered in the past, and which is presented as a reference next to the current price.

It may be labeled as the original price, the list price, or the maximum retail price for which the product is sold.

See `price` for format details.

If `regularPrice` is not None, it should always be higher than `price`.

url: *Optional*[str]

Main URL from which the product data could be extracted.

class `zyte_common_items.ProductListMetadata(**kwargs)`

Metadata class for `zyte_common_items.ProductList.metadata`.

dateDownloaded: *Optional*[str]

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

6.1.3 Product navigation

class `zyte_common_items.ProductNavigation(**kwargs)`

Represents the navigational aspects of a product listing page on an e-commerce website

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → `Optional[float]`

Returns the item probability if available, otherwise None.

categoryName: `Optional[str]`

Name of the category/page with the product list.

Format:

- trimmed (no whitespace at the beginning or the end of the description string)

items: `Optional[List[ProbabilityRequest]]`

List of product links found on the page category ordered by their position in the page.

metadata: `Optional[ProductNavigationMetadata]`

Data extraction process metadata.

nextPage: `Optional[Request]`

A link to the next page, if available.

pageNumber: `Optional[int]`

Number of the current page.

It should only be extracted if the webpage shows a page number.

It must be 1-based. For example, if the first page of a listing is numbered as 0 on the website, it should be extracted as 1 nonetheless.

subCategories: `Optional[List[ProbabilityRequest]]`

List of sub-category links ordered by their position in the page.

url: `str`

Main URL from which the data is extracted.

class `zyte_common_items.ProductNavigationMetadata(**kwargs)`

Metadata class for `zyte_common_items.ProductNavigation.metadata`.

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

6.1.4 Article

class `zyte_common_items.Article(**kwargs)`

Article, typically seen on online news websites, blogs, or announcement sections.

`url` is the only required attribute.

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → *Optional[float]*

Returns the item probability if available, otherwise `None`.

articleBody: *Optional[str]*

Clean text of the article, including sub-headings, with newline separators.

Format:

- trimmed (no whitespace at the beginning or the end of the body string),
- line breaks included,
- no length limit,
- no normalization of Unicode characters.

articleBodyHtml: *Optional[str]*

Simplified and standardized HTML of the article, including sub-headings, image captions and embedded content (videos, tweets, etc.).

Format: HTML string normalized in a consistent way.

audios: *Optional[List[Audio]]*

All audios.

authors: *Optional[List[Author]]*

All authors of the article.

breadcrumbs: *Optional[List[Breadcrumb]]*

Webpage breadcrumb trail.

canonicalUrl: *Optional[str]*

Canonical form of the URL, as indicated by the website.

See also `url`.

dateModified: *Optional[str]*

Date when the article was most recently modified.

Format: ISO 8601 format: “YYYY-MM-DDThh:mm:ssZ” or “YYYY-MM-DDThh:mm:ss±zz:zz”.

With timezone, if available.

dateModifiedRaw: *Optional[str]*

Same date as *dateModified*, but :before parsing/normalization, i.e. as it appears on the website.

datePublished: `Optional[str]`

Publication date of the article.

Format: ISO 8601 format: “YYYY-MM-DDThh:mm:ssZ” or “YYYY-MM-DDThh:mm:ss±zz:zz”.

With timezone, if available.

If the actual publication date is not found, the value of `dateModified` is used instead.

datePublishedRaw: `Optional[str]`

Same date as `datePublished`, but :before parsing/normalization, i.e. as it appears on the website.

description: `Optional[str]`

A short summary of the article.

It can be either human-provided (if available), or auto-generated.

headline: `Optional[str]`

Headline or title.

images: `Optional[List[Image]]`

All images.

inLanguage: `Optional[str]`

Language of the article, as an ISO 639-1 language code.

Sometimes the article language is not the same as the web page overall language.

mainImage: `Optional[Image]`

Main image.

metadata: `Optional[ArticleMetadata]`

Data extraction process metadata.

url: `str`

The main URL of the article page.

The URL of the final response, after any redirects.

Required attribute.

In case there is no article data on the page or the page was not reached, the returned “empty” item would still contain this URL field.

videos: `Optional[List[Video]]`

All videos.

class `zyte_common_items.ArticleMetadata(**kwargs)`

Metadata class for `zyte_common_items.Article.metadata`.

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

probability: `Optional[float]`

The probability (0 for 0%, 1 for 100%) that the resource features the expected data type.

For example, if the extraction of a product from a given URL is requested, and that URL points to the webpage of a product with complete certainty, the value should be `1`. If with complete certainty the webpage features a job listing instead of a product, the value should be `0`. When there is no complete certainty, the value could be anything in between (e.g. `0.96`).

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

6.1.5 Article list

class `zyte_common_items.ArticleList(**kwargs)`

Article list from an article listing page.

The `url` attribute is the only required attribute, all other fields are optional.

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → `Optional[float]`

Returns the item probability if available, otherwise None.

articles: `Optional[List[ArticleFromList]]`

List of article details found on the page.

The order of the articles reflects their position on the page.

breadcrumbs: `Optional[List[Breadcrumb]]`

Webpage breadcrumb trail.

canonicalUrl: `Optional[str]`

Canonical form of the URL, as indicated by the website.

See also `url`.

metadata: `Optional[ArticleListMetadata]`

Data extraction process metadata.

url: `str`

The main URL of the article list.

The URL of the final response, after any redirects.

Required attribute.

In case there is no article list data on the page or the page was not reached, the returned item still contain this URL field and all the other available datapoints.

class `zyte_common_items.ArticleFromList(**kwargs)`

Article from an article list from an article listing page.

See `ArticleList`.

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → `Optional[float]`

Returns the item probability if available, otherwise None.

articleBody: `Optional[str]`

Clean text of the article, including sub-headings, with newline separators.

Format:

- trimmed (no whitespace at the beginning or the end of the body string),
- line breaks included,
- no length limit,
- no normalization of Unicode characters.

authors: `Optional[List[Author]]`

All authors of the article.

datePublished: `Optional[str]`

Publication date of the article.

Format: ISO 8601 format: “YYYY-MM-DDThh:mm:ssZ” or “YYYY-MM-DDThh:mm:ss±zz:zz”.

With timezone, if available.

If the actual publication date is not found, the date of the last modification is used instead.

datePublishedRaw: `Optional[str]`

Same date as *datePublished*, but :before parsing/normalization, i.e. as it appears on the website.

headline: `Optional[str]`

Headline or title.

images: `Optional[List[Image]]`

All images.

inLanguage: `Optional[str]`

Language of the article, as an ISO 639-1 language code.

Sometimes the article language is not the same as the web page overall language.

mainImage: `Optional[Image]`

Main image.

metadata: `Optional[ProbabilityMetadata]`

Data extraction process metadata.

url: `Optional[str]`

Main URL.

class `zyte_common_items.ArticleListMetadata(**kwargs)`

Metadata class for *zyte_common_items.ArticleList.metadata*.

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

6.1.6 Article navigation

class `zyte_common_items.ArticleNavigation(**kwargs)`

Represents the navigational aspects of an article listing webpage.

See [ArticleList](#).

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → `Optional[float]`

Returns the item probability if available, otherwise `None`.

categoryName: `Optional[str]`

Name of the category/page.

Format:

- trimmed (no whitespace at the beginning or the end of the description string)

items: `Optional[List[ProbabilityRequest]]`

Links to listed items in order of appearance.

metadata: `Optional[ArticleNavigationMetadata]`

Data extraction process metadata.

nextPage: `Optional[Request]`

A link to the next page, if available.

pageNumber: `Optional[int]`

Number of the current page.

It should only be extracted if the webpage shows a page number.

It must be 1-based. For example, if the first page of a listing is numbered as 0 on the website, it should be extracted as 1 nonetheless.

subCategories: `Optional[List[ProbabilityRequest]]`

List of sub-category links ordered by their position in the page.

url: `str`

Main URL from which the data is extracted.

class `zyte_common_items.ArticleNavigationMetadata(**kwargs)`

Metadata class for `zyte_common_items.ArticleNavigation.metadata`.

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

6.1.7 Business place

```
class zyte_common_items.BusinessPlace(**kwargs)
```

Business place, with properties typically seen on maps or business listings.

`url` is the only required attribute.

```
classmethod from_dict(item: Optional[Dict])
```

Read an item from a dictionary.

```
classmethod from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List
```

Read items from a list.

```
get_probability() → Optional[float]
```

Returns the item probability if available, otherwise `None`.

```
actions: Optional[List[NamedLink]]
```

List of actions that can be performed directly from the URLs on the place page, including URLs.

```
additionalProperties: Optional[List[AdditionalProperty]]
```

List of name-value pairs of any unmapped additional properties specific to the place.

```
address: Optional[Address]
```

The address details of the place.

```
aggregateRating: Optional[AggregateRating]
```

The overall rating, based on a collection of reviews or ratings.

```
amenityFeatures: Optional[List[Amenity]]
```

List of amenities of the place.

```
categories: Optional[List[str]]
```

List of categories the place belongs to.

```
containedInPlace: Optional[ParentPlace]
```

If the place is located inside another place, these are the details of the parent place.

```
description: Optional[str]
```

The description of the place.

Stripped of white spaces.

```
features: Optional[List[str]]
```

List of frequently mentioned features of this place.

```
images: Optional[List[Image]]
```

A list of URL values of all images of the place.

```
isVerified: Optional[bool]
```

If the information is verified by the owner of this place.

```
map: Optional[str]
```

URL to a map of the place.

```
metadata: Optional[BusinessPlaceMetadata]
```

Data extraction process metadata.

```
name: Optional[str]
```

The name of the place.

openingHours: `Optional[List[OpeningHoursItem]]`

Ordered specification of opening hours, including data for opening and closing time for each day of the week.

placeId: `Optional[str]`

Unique identifier of the place on the website.

priceRange: `Optional[str]`

How is the price range of the place viewed by its customers (from z to zzzz).

reservationAction: `Optional[NamedLink]`

The details of the reservation action, e.g. table reservation in case of restaurants or room reservation in case of hotels.

reviewSites: `Optional[List[NamedLink]]`

List of partner review sites.

starRating: `Optional[StarRating]`

Official star rating of the place.

tags: `Optional[List[str]]`

List of the tags associated with the place.

telephone: `Optional[str]`

The phone number associated with the place, as it appears on the page.

timezone: `Optional[str]`

Which timezone is the place situated in.

Standard: Name compliant with IANA tz database (tzdata).

url: `Optional[str]`

The main URL that the place data was extracted from.

The URL of the final response, after any redirects.

In case there is no product data on the page or the page was not reached, the returned “empty” item would still contain url field and metadata field with dateDownloaded.

website: `Optional[str]`

The URL pointing to the official website of the place.

class `zyte_common_items.BusinessPlaceMetadata(**kwargs)`

Metadata class for `zyte_common_items.BusinessPlace.metadata`.

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

probability: `Optional[float]`

The probability (0 for 0%, 1 for 100%) that the resource features the expected data type.

For example, if the extraction of a product from a given URL is requested, and that URL points to the webpage of a product with complete certainty, the value should be 1. If with complete certainty the webpage features a job listing instead of a product, the value should be 0. When there is no complete certainty, the value could be anything in between (e.g. 0.96).

searchText: `Optional[str]`

The search text used to find the item.

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

6.1.8 Real estate

class `zyte_common_items.RealEstate(**kwargs)`

Real state offer, typically seen on real estate offer aggregator websites.

`url` is the only required attribute.

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → `Optional[float]`

Returns the item probability if available, otherwise `None`.

additionalProperties: `Optional[List[AdditionalProperty]]`

A name-value pair field holding information pertaining to specific features. Usually in a form of a specification table or freeform specification list.

address: `Optional[Address]`

The details of the address of the real estate.

area: `Optional[RealEstateArea]`

Real estate area details.

breadcrumbs: `Optional[List[Breadcrumb]]`

Webpage breadcrumb trail.

currency: `Optional[str]`

The currency of the price, in 3-letter ISO 4217 format.

currencyRaw: `Optional[str]`

Currency associated with the price, as appears on the page (no post-processing).

datePublished: `Optional[str]`

Publication date of the real estate offer.

Format: ISO 8601 format: “YYYY-MM-DDThh:mm:ssZ”

With timezone, if available.

datePublishedRaw: `Optional[str]`

Same date as `datePublished`, but before parsing/normalization, i.e. as it appears on the website.

description: `Optional[str]`

The description of the real estate.

Format:

- trimmed (no whitespace at the beginning or the end of the description string),
- line breaks included,
- no length limit,
- no normalization of Unicode characters,

- no concatenation of description from different parts of the page.

images: `Optional[List[Image]]`

A list of URL values of all images of the real estate.

mainImage: `Optional[Image]`

The details of the main image of the real estate.

metadata: `Optional[RealEstateMetadata]`

Contains metadata about the data extraction process.

name: `Optional[str]`

The name of the real estate.

numberOfBathroomsTotal: `Optional[int]`

The total number of bathrooms in the real estate.

numberOfBedrooms: `Optional[int]`

The number of bedrooms in the real estate.

numberOfFullBathrooms: `Optional[int]`

The number of full bathrooms in the real estate.

numberOfPartialBathrooms: `Optional[int]`

The number of partial bathrooms in the real estate.

numberOfRooms: `Optional[int]`

The number of rooms (excluding bathrooms and closets) of the real estate.

price: `Optional[str]`

The offer price of the real estate.

propertyType: `Optional[str]`

Type of the property, e.g. flat, house, land.

realEstateId: `Optional[str]`

The identifier of the real estate, usually assigned by the seller and unique within a website, similar to product SKU.

rentalPeriod: `Optional[str]`

The rental period to which the rental price applies, only available in case of rental. Usually weekly, monthly, quarterly, yearly.

tradeType: `Optional[str]`

Type of a trade action: buying or renting.

url: `str`

The url of the final response, after any redirects.

virtualTourUrl: `Optional[str]`

The URL of the virtual tour of the real estate.

yearBuilt: `Optional[int]`

The year the real estate was built.

class `zyte_common_items.RealEstateMetadata(**kwargs)`

Metadata class for `zyte_common_items.RealEstate.metadata`.

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

probability: `Optional[float]`

The probability (0 for 0%, 1 for 100%) that the resource features the expected data type.

For example, if the extraction of a product from a given URL is requested, and that URL points to the webpage of a product with complete certainty, the value should be *1*. If with complete certainty the webpage features a job listing instead of a product, the value should be *0*. When there is no complete certainty, the value could be anything in between (e.g. *0.96*).

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

6.1.9 Job posting

class `zyte_common_items.JobPosting(**kwargs)`

A job posting, typically seen on job posting websites or websites of companies that are hiring.

`url` is the only required attribute.

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → `Optional[float]`

Returns the item probability if available, otherwise `None`.

baseSalary: `Optional[BaseSalary]`

The base salary of the job or of an employee in the proposed role.

dateModified: `Optional[str]`

The date when the job posting was most recently modified.

Format: ISO 8601 format: “YYYY-MM-DDThh:mm:ssZ”

With timezone, if available.

dateModifiedRaw: `Optional[str]`

Same date as `dateModified`, but before parsing/normalization, i.e. as it appears on the website.

datePublished: `Optional[str]`

Publication date of the job posting.

Format: ISO 8601 format: “YYYY-MM-DDThh:mm:ssZ”

With timezone, if available.

datePublishedRaw: `Optional[str]`

Same date as `datePublished`, but before parsing/normalization, i.e. as it appears on the website.

description: `Optional[str]`

A description of the job posting including sub-headings, with newline separators.

Format:

- trimmed (no whitespace at the beginning or the end of the description string),
- line breaks included,
- no length limit,
- no normalization of Unicode characters.

descriptionHtml: `Optional[str]`

Simplified HTML of the description, including sub-headings, image captions and embedded content.

employmentType: `Optional[str]`

Type of employment (e.g. full-time, part-time, contract, temporary, seasonal, internship).

headline: `Optional[str]`

The headline of the job posting.

hiringOrganization: `Optional[HiringOrganization]`

Information about the organization offering the job position.

jobLocation: `Optional[JobLocation]`

A (typically single) geographic location associated with the job position.

jobPostingId: `Optional[str]`

The identifier of the job posting.

jobStartDate: `Optional[str]`

Job start date

Format: ISO 8601 format: “YYYY-MM-DDThh:mm:ssZ”

With timezone, if available.

jobStartDateRaw: `Optional[str]`

Same date as jobStartDate, but before parsing/normalization, i.e. as it appears on the website.

jobTitle: `Optional[str]`

The title of the job posting.

metadata: `Optional[JobPostingMetadata]`

Contains metadata about the data extraction process.

remoteStatus: `Optional[str]`

Specifies the remote status of the position.

requirements: `Optional[List[str]]`

Candidate requirements for the job.

url: `str`

The url of the final response, after any redirects.

validThrough: `Optional[str]`

The date after which the job posting is not valid, e.g. the end of an offer.

Format: ISO 8601 format: “YYYY-MM-DDThh:mm:ssZ”

With timezone, if available.

validThroughRaw: `Optional[str]`

Same date as validThrough, but before parsing/normalization, i.e. as it appears on the website.


```
class zyte_common_items.JobPostingMetadata(**kwargs)
```

Metadata class for `zyte_common_items.JobPosting.metadata`.

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

probability: `Optional[float]`

The probability (0 for 0%, 1 for 100%) that the resource features the expected data type.

For example, if the extraction of a product from a given URL is requested, and that URL points to the webpage of a product with complete certainty, the value should be *1*. If with complete certainty the webpage features a job listing instead of a product, the value should be *0*. When there is no complete certainty, the value could be anything in between (e.g. *0.96*).

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

6.1.10 Social media post

```
class zyte_common_items.SocialMediaPost(**kwargs)
```

Represents a single social media post.

classmethod from_dict(item: `Optional[Dict]`)

Read an item from a dictionary.

classmethod from_list(items: `Optional[List[Dict]]`, *, trail: `Optional[str] = None`) → `List`

Read items from a list.

get_probability() → `Optional[float]`

Returns the item probability if available, otherwise None.

author: `Optional[SocialMediaPostAuthor]`

Details of the author of the post.

No easily identifiable information can be contained in here, such as usernames.

datePublished: `Optional[str]`

The timestamp at which the post was created.

Format: Timezone: UTC. ISO 8601 format: “YYYY-MM-DDThh:mm:ssZ”

hashtags: `Optional[List[str]]`

The list of hashtags contained in the post.

mediaUrls: `Optional[List[Url]]`

The list of URLs of media files (images, videos, etc.) linked from the post.

metadata: `Optional[SocialMediaPostMetadata]`

Contains metadata about the data extraction process.

postId: `Optional[str]`

The identifier of the post.

reactions: `Optional[Reactions]`

Details of reactions to the post.

text: `Optional[str]`

The text content of the post.

url: `str`

The URL of the final response, after any redirects.

class `zyte_common_items.SocialMediaPostMetadata(**kwargs)`

Metadata class for `zyte_common_items.SocialMediaPost.metadata`.

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

probability: `Optional[float]`

The probability (0 for 0%, 1 for 100%) that the resource features the expected data type.

For example, if the extraction of a product from a given URL is requested, and that URL points to the webpage of a product with complete certainty, the value should be *1*. If with complete certainty the webpage features a job listing instead of a product, the value should be *0*. When there is no complete certainty, the value could be anything in between (e.g. *0.96*).

searchText: `Optional[str]`

The search text used to find the item.

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

6.1.11 Search Request templates

class `zyte_common_items.SearchRequestTemplate(**kwargs)`

Request template to build a search *Request*.

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → `Optional[float]`

Returns the item probability if available, otherwise *None*.

request(*, keyword: *str*) → *Request*

Return a *Request* to search for *keyword*.

body: `Optional[str]`

Jinja template for *Request.body*.

It must be a plain *str*, not *bytes* or a Base64-encoded *str*. Base64-encoding is done by `request()` after rendering this value as a Jinja template.

Defining a non-UTF-8 body is not supported.

headers: `Optional[List[Header]]`

List of *Header*, for *Request.headers*, where every *name* and *value* is a Jinja template.

When a header name template renders into an empty string (after stripping spacing), that header is removed from the resulting list of headers.

metadata: `Optional[SearchRequestTemplateMetadata]`

Data extraction process metadata.

method: `str`

Jinja template for `Request.method`.

url: `str`

Jinja template for `Request.url`.

class `zyte_common_items.SearchRequestTemplateMetadata(**kwargs)`

Metadata class for `zyte_common_items.SearchRequestTemplate.metadata`.

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

probability: `Optional[float]`

The probability (0 for 0%, 1 for 100%) that the resource features the expected data type.

For example, if the extraction of a product from a given URL is requested, and that URL points to the webpage of a product with complete certainty, the value should be `1`. If with complete certainty the webpage features a job listing instead of a product, the value should be `0`. When there is no complete certainty, the value could be anything in between (e.g. `0.96`).

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

6.1.12 Custom items

Subclass `Item` to create your own item classes.

class `zyte_common_items.base.ProbabilityMixin(**kwargs)`

Provides `get_probability()` to make it easier to access the probability of an item or item component that is nested under its metadata attribute.

get_probability() → `Optional[float]`

Returns the item probability if available, otherwise `None`.

class `zyte_common_items.Item(**kwargs)`

Base class for `items`.

_unknown_fields_dict: `dict`

Contains unknown attributes fed into the item through `from_dict()` or `from_list()`.

classmethod `from_dict(item: Optional[Dict])`

Read an item from a dictionary.

classmethod `from_list(items: Optional[List[Dict]], *, trail: Optional[str] = None) → List`

Read items from a list.

get_probability() → `Optional[float]`

Returns the item probability if available, otherwise `None`.

6.2 Page object API

6.2.1 Product

```
class zyte_common_items.BaseProductPage(**kwargs)
    Bases: BasePage, DescriptionMixin, PriceMixin, Returns[Product],
           HasMetadata[ProductMetadata]
    BasePage subclass for Product.

class zyte_common_items.ProductPage(**kwargs)
    Bases: Page, DescriptionMixin, PriceMixin, Returns[Product], HasMetadata[ProductMetadata]
    Page subclass for Product.

class zyte_common_items.AutoProductPage(**kwargs)
    Bases: BaseProductPage
```

6.2.2 Product list

```
class zyte_common_items.BaseProductListPage(**kwargs)
    Bases: BasePage, Returns[ProductList], HasMetadata[ProductListMetadata]
    BasePage subclass for ProductList.

class zyte_common_items.ProductListPage(**kwargs)
    Bases: Page, Returns[ProductList], HasMetadata[ProductListMetadata]
    Page subclass for ProductList.

class zyte_common_items.AutoProductListPage(**kwargs)
    Bases: BaseProductListPage
```

6.2.3 Product navigation

```
class zyte_common_items.BaseProductNavigationPage(**kwargs)
    Bases: BasePage, Returns[ProductNavigation], HasMetadata[ProductNavigationMetadata]
    BasePage subclass for ProductNavigation.

class zyte_common_items.ProductNavigationPage(**kwargs)
    Bases: Page, Returns[ProductNavigation], HasMetadata[ProductNavigationMetadata]
    Page subclass for ProductNavigation.

class zyte_common_items.AutoProductNavigationPage(**kwargs)
    Bases: BaseProductNavigationPage
```

6.2.4 Article

```
class zyte_common_items.BaseArticlePage(**kwargs)
    Bases: BasePage, Returns[Article], HasMetadata[ArticleMetadata]
    BasePage subclass for Article.

class zyte_common_items.ArticlePage(**kwargs)
    Bases: Page, Returns[Article], HasMetadata[ArticleMetadata]
    Page subclass for Article.

class zyte_common_items.AutoArticlePage(**kwargs)
    Bases: BaseArticlePage
```

6.2.5 Article list

```
class zyte_common_items.BaseArticleListPage(**kwargs)
    Bases: BasePage, Returns[ArticleList], HasMetadata[ArticleListMetadata]
    BasePage subclass for ArticleList.

class zyte_common_items.ArticleListPage(**kwargs)
    Bases: Page, Returns[ArticleList], HasMetadata[ArticleListMetadata]
    Page subclass for ArticleList.

class zyte_common_items.AutoArticleListPage(**kwargs)
    Bases: BaseArticleListPage
```

6.2.6 Article navigation

```
class zyte_common_items.BaseArticleNavigationPage(**kwargs)
    Bases: BasePage, Returns[ArticleNavigation], HasMetadata[ArticleNavigationMetadata]
    BasePage subclass for ArticleNavigation.

class zyte_common_items.ArticleNavigationPage(**kwargs)
    Bases: Page, Returns[ArticleNavigation], HasMetadata[ArticleNavigationMetadata]
    Page subclass for ArticleNavigation.

class zyte_common_items.AutoArticleNavigationPage(**kwargs)
    Bases: BaseArticleNavigationPage
```

6.2.7 Business place

```
class zyte_common_items.BaseBusinessPlacePage(**kwargs)
    Bases: BasePage, Returns[BusinessPlace], HasMetadata[BusinessPlaceMetadata]
    BasePage subclass for BusinessPlace.
```

```
class zyte_common_items.BusinessPlacePage(**kwargs)
    Bases: Page, Returns[BusinessPlace], HasMetadata[BusinessPlaceMetadata]
    Page subclass for BusinessPlace.

class zyte_common_items.AutoBusinessPlacePage(**kwargs)
    Bases: BaseBusinessPlacePage
```

6.2.8 Real estate

```
class zyte_common_items.BaseRealEstatePage(**kwargs)
    Bases: BasePage, Returns[RealEstate], HasMetadata[RealEstateMetadata]
    BasePage subclass for RealEstate.

class zyte_common_items.RealEstatePage(**kwargs)
    Bases: Page, Returns[RealEstate], HasMetadata[RealEstateMetadata]
    Page subclass for RealEstate.

class zyte_common_items.AutoRealEstatePage(**kwargs)
    Bases: BaseRealEstatePage
```

6.2.9 Job posting

```
class zyte_common_items.BaseJobPostingPage(**kwargs)
    Bases: BasePage, DescriptionMixin, Returns[JobPosting], HasMetadata[JobPostingMetadata]
    BasePage subclass for JobPosting.

class zyte_common_items.JobPostingPage(**kwargs)
    Bases: Page, DescriptionMixin, Returns[JobPosting], HasMetadata[JobPostingMetadata]
    Page subclass for JobPosting.

class zyte_common_items.AutoJobPostingPage(**kwargs)
    Bases: BaseJobPostingPage
```

6.2.10 Social media post

```
class zyte_common_items.BaseSocialMediaPostPage(**kwargs)
    Bases: BasePage, Returns[SocialMediaPost], HasMetadata[SocialMediaPostMetadata]

class zyte_common_items.SocialMediaPostPage(**kwargs)
    Bases: Page, Returns[SocialMediaPost], HasMetadata[SocialMediaPostMetadata]

class zyte_common_items.AutoSocialMediaPostPage(**kwargs)
    Bases: BaseSocialMediaPostPage
```

6.2.11 Request templates

```
class zyte_common_items.SearchRequestTemplatePage(**kwargs)
    Bases: ItemPage[SearchRequestTemplate], HasMetadata[SearchRequestTemplateMetadata]
```

6.2.12 Mixins

```
class zyte_common_items.pages.DescriptionMixin
    Provides description and descriptionHtml field implementations.

    description: str
        Plain-text description. The default implementation makes it from the descriptionHtml field if that is
        user-defined.

    descriptionHtml: str
        HTML description. The default implementation makes it from the description field if that is user-
        defined.

class zyte_common_items.pages.PriceMixin
    Provides price-related field implementations.

    currency: str
        Price currency ISO 4217 alphabetic code (e.g. "USD"). The default implementation returns self.
        CURRENCY if this attribute is defined.

    currencyRaw: str
        Price currency as it appears on the webpage (no post-processing), e.g. "$". The default implementation
        uses the data extracted by price_processor() from the price field.
```

6.2.13 Custom page objects

Subclass *Page* to create your own page object classes that rely on *HttpResponse*.

If you do not want *HttpResponse* as input, you can inherit from *BasePage* instead.

Your subclasses should also inherit generic classes *web_poet.pages>Returns* and *zyte_common_items.HasMetadata* to indicate their item and metadata classes.

```
class zyte_common_items.pages.base._BasePage(**kwargs)

class zyte_common_items.BasePage(**kwargs)
    Bases: _BasePage

    Base class for page object classes that has RequestUrl as a dependency.

    metadata
        Data extraction process metadata.

        dateDownloaded is set to the current UTC date and time.

        probability is set to 1.0.

    url: str
        Main URL from which the data has been extracted.
```

no_item_found() → ItemT

Return an item with the current url and probability=0, indicating that the passed URL doesn't contain the expected item.

Use it in your `.validate_input` implementation.

class `zyte_common_items.Page`(**kwargs)

Bases: `_BasePage`, `WebPage`

Base class for page object classes that has `HttpResponse` as a dependency.

metadata: `zyte_common_items.Metadata`

Data extraction process metadata.

`dateDownloaded` is set to the current UTC date and time.

`probability` is set to 1.0.

url: `str`

Main URL from which the data has been extracted.

no_item_found() → ItemT

Return an item with the current url and probability=0, indicating that the passed URL doesn't contain the expected item.

Use it in your `.validate_input` implementation.

class `zyte_common_items.HasMetadata`

Inherit from this generic mixin to set the metadata class used by a page class.

6.3 Extractor API

API reference of provided *extractors*.

6.3.1 Product from list

class `zyte_common_items.ProductFromListExtractor`

`Extractor` for *ProductFromList*.

class `zyte_common_items.ProductFromListSelectorExtractor`(selector: *Selector*)

`SelectorExtractor` for *ProductFromList*.

6.3.2 Product variant

class `zyte_common_items.ProductVariantExtractor`

`Extractor` for *ProductVariant*.

class `zyte_common_items.ProductVariantSelectorExtractor`(selector: *Selector*)

`SelectorExtractor` for *ProductVariant*.

6.4 Field processor API

API reference of provided *field processors*.

6.4.1 Built-in field processors

`zyte_common_items.processors.brand_processor(value: Union[Selector, HtmlElement], page: Any) → Any`

Convert the data into a brand name if possible.

Supported inputs are `Selector`, `SelectorList` and `HtmlElement`. Other inputs are returned as is.

`zyte_common_items.processors.breadcrumbs_processor(value: Any, page: Any) → Any`

Convert the data into a list of `Breadcrumb` objects if possible.

Supported inputs are `Selector`, `SelectorList`, `HtmlElement` and an iterable of `zyte_parsers.Breadcrumb` objects. Other inputs are returned as is.

`zyte_common_items.processors.description_processor(value: Any, page: Any) → Any`

Convert the data into a cleaned up text if possible.

Uses the `clear-html` library.

Supported inputs are `Selector`, `SelectorList` and `HtmlElement`. Other inputs are returned as is.

Puts the cleaned `HtmlElement` object into `page._description_node` and the cleaned text into `page._description_str`.

`zyte_common_items.processors.description_html_processor(value: Union[Selector, HtmlElement], page: Any) → Any`

Convert the data into a cleaned up HTML if possible.

Uses the `clear-html` library.

Supported inputs are `Selector`, `SelectorList` and `HtmlElement`. Other inputs are returned as is.

Puts the cleaned `HtmlElement` object into `page._descriptionHtml_node`.

`zyte_common_items.processors.gtin_processor(value: Union[SelectorList, Selector, HtmlElement, str], page: Any) → Any`

Convert the data into a list of `Gtin` objects if possible.

Supported inputs are `str`, `Selector`, `SelectorList`, `HtmlElement`, an iterable of `str` and an iterable of `zyte_parsers.Gtin` objects. Other inputs are returned as is.

`zyte_common_items.processors.price_processor(value: Union[Selector, HtmlElement], page: Any) → Any`

Convert the data into a price string if possible.

Uses the `price-parser` library.

Supported inputs are `Selector`, `SelectorList` and `HtmlElement`. Other inputs are returned as is.

Puts the parsed `Price` object into `page._parsed_price`.

`zyte_common_items.processors.rating_processor(value: Any, page: Any) → Any`

Convert the data into an `AggregateRating` object if possible.

Supported inputs are selector-like objects (`Selector`, `SelectorList`, or `HtmlElement`).

The input can also be a dictionary with one or more of the `AggregateRating` fields as keys. The values for those keys can be either final values, to be assigned to the corresponding fields, or selector-like objects.

If a returning dictionary is missing the `bestRating` field and `ratingValue` is a selector-like object, `bestRating` may be extracted.

For example, for the following input HTML:

```
<span class="rating">3.8 out of 5 stars</span>
<a class="reviews">See all 7 reviews</a>
```

You can use:

```
@field
def aggregateRating(self):
    return {
        "ratingValue": self.css(".rating"),
        "reviewCount": self.css(".reviews"),
    }
```

To get:

```
AggregateRating(
    bestRating=5.0,
    ratingValue=3.8,
    reviewCount=7,
)
```

`zyte_common_items.processors.simple_price_processor`(*value: Union[Selector, HtmlElement], page: Any*) → *Any*

Convert the data into a price string if possible.

Uses the [price-parser](#) library.

Supported inputs are [Selector](#), [SelectorList](#) and [HtmlElement](#). Other inputs are returned as is.

6.5 Components

These classes are used to map data within *items*, and are not tied to any specific item type.

class `zyte_common_items.AdditionalProperty`(***kwargs*)

A name-value pair.

See [Product.additionalProperties](#).

name: `str`

Name.

value: `str`

Value.

class `zyte_common_items.Address`(***kwargs*)

Address item.

addressCity: `Optional[str]`

The city the place is located in.

addressCountry: `Optional[str]`

The country the place is located in.

The country name or the ISO 3166-1 alpha-2 country code.

addressLocality: `Optional[str]`

The locality to which the place belongs.

addressRaw: `Optional[str]`

The raw address information, as it appears on the website.

addressRegion: `Optional[str]`

The region of the place.

latitude: `Optional[float]`

Geographical latitude of the place.

longitude: `Optional[float]`

Geographical longitude of the place.

postalCode: `Optional[str]`

The postal code of the address.

postalCodeAux: `Optional[str]`

The auxiliary part of the postal code.

It may include a state abbreviation or town name, depending on local standards.

streetAddress: `Optional[str]`

The street address of the place.

class `zyte_common_items.AggregateRating(**kwargs)`

Aggregate data about reviews and ratings.

At least one of *ratingValue* or *reviewCount* is required.

See *Product.aggregateRating*.

bestRating: `Optional[float]`

Maximum value of the rating system.

ratingValue: `Optional[float]`

Average value of all ratings.

reviewCount: `Optional[int]`

Review count.

class `zyte_common_items.Amenity(**kwargs)`

An amenity that a business place has

name: `str`

Name of amenity.

value: `bool`

Availability of the amenity.

class `zyte_common_items.Audio(**kwargs)`

Audio.

See *Article.audios*.

url: `str`

URL.

When multiple URLs exist for a given media element, pointing to different-quality versions, the highest-quality URL should be used.

Data URIs are not allowed in this attribute.

class `zyte_common_items.Author(**kwargs)`

Author of an article.

See [`Article.authors`](#).

email: `Optional[str]`

Email.

name: `Optional[str]`

Full name.

nameRaw: `Optional[str]`

Text from which [`name`](#) was extracted.

url: `Optional[str]`

URL of the details page of the author.

class `zyte_common_items.BaseSalary(**kwargs)`

Base salary of a job offer.

currency: `Optional[str]`

Currency associated with the salary amount.

currencyRaw: `Optional[str]`

Currency associated with the salary amount, without normalization.

rateType: `Optional[str]`

The type of rate associated with the salary, e.g. monthly, annual, daily.

raw: `Optional[str]`

Salary amount as it appears on the website.

valueMax: `Optional[str]`

The maximum value of the base salary as a number string.

valueMin: `Optional[str]`

The minimum value of the base salary as a number string.

class `zyte_common_items.Brand(**kwargs)`

Brand.

See [`Product.brand`](#).

name: `str`

Name as it appears on the source webpage (no post-processing).

class `zyte_common_items.Breadcrumb(**kwargs)`

A breadcrumb from the [`breadcrumb`](#) trail of a webpage.

See [`Product.breadcrumbs`](#).

name: `Optional[str]`
 Displayed name.

url: `Optional[str]`
 Target URL.

class `zyte_common_items.Gtin(**kwargs)`
 GTIN type-value pair.
 See [Product.gtin](#).

type: `str`
 Identifier of the GTIN format of value.
 One of: "gtin13", "gtin8", "gtin14", "isbn10", "isbn13", "ismn", "issn", "upc".

value: `str`
 Value.
 It should only contain digits.

class `zyte_common_items.Header(**kwargs)`
 An HTTP header

name: `str`
 Name of the header

value: `str`
 Value of the header

class `zyte_common_items.HiringOrganization(**kwargs)`
 Organization that is hiring for a job offer.

id: `Optional[str]`
 Identifier of the organization used by job posting website.

name: `Optional[str]`
 Name of the hiring organization.

nameRaw: `Optional[str]`
 Organization information as available on the website.

class `zyte_common_items.Image(**kwargs)`
 Image.
 See for example [Product.images](#) and [Product.mainImage](#).

url: `str`
 URL.
 When multiple URLs exist for a given media element, pointing to different-quality versions, the highest-quality URL should be used.
 Data URIs are not allowed in this attribute.

class `zyte_common_items.JobLocation(**kwargs)`
 Location of a job offer.

raw: `Optional[str]`
 Job location, as it appears on the website.

```
class zyte_common_items.Link(**kwargs)
```

A link from a webpage to another webpage.

```
text: Optional[str]
```

Displayed text.

```
url: Optional[str]
```

Target URL.

```
class zyte_common_items.NamedLink(**kwargs)
```

A link from a webpage to another webpage.

```
name: Optional[str]
```

The name of the link.

```
url: Optional[str]
```

Target URL.

```
class zyte_common_items.OpeningHoursItem(**kwargs)
```

Specification of opening hours of a business place.

```
closes: Optional[str]
```

Closing time in ISO 8601 format, local time.

```
dayOfWeek: Optional[str]
```

English weekday name.

```
opens: Optional[str]
```

Opening time in ISO 8601 format, local time.

```
rawCloses: Optional[str]
```

Closing time, as it appears on the page, without processing.

```
rawDayOfWeek: Optional[str]
```

Day of the week, as it appears on the page, without processing.

```
rawOpens: Optional[str]
```

Opening time, as it appears on the page, without processing.

```
class zyte_common_items.ParentPlace(**kwargs)
```

If the place is located inside another place, these are the details of the parent place.

```
name: str
```

Name of the parent place.

```
placeId: str
```

Identifier of the parent place.

```
class zyte_common_items.ProbabilityMetadata(**kwargs)
```

Data extraction process metadata.

```
probability: Optional[float]
```

The probability (0 for 0%, 1 for 100%) that the resource features the expected data type.

For example, if the extraction of a product from a given URL is requested, and that URL points to the webpage of a product with complete certainty, the value should be *1*. If with complete certainty the webpage features a job listing instead of a product, the value should be *0*. When there is no complete certainty, the value could be anything in between (e.g. *0.96*).

```

class zyte_common_items.ProbabilityRequest(**kwargs)
    A Request that includes a probability value.

    metadata: Optional[ProbabilityMetadata]
        Data extraction process metadata.

class zyte_common_items.Reactions(**kwargs)
    Details of reactions to a post.

    dislikes: Optional[int]
        Number of dislikes or other negative reactions to the post.

    likes: Optional[int]
        Number of likes or other positive reactions to the post.

    reposts: Optional[int]
        Number of times the post has been shared.

class zyte_common_items.RealEstateArea(**kwargs)
    Area of a place, with type, units, value and raw value.

    areaType: Optional[str]
        Type of area, one of: LOT, FLOOR

    raw: str
        Area in the raw format, as it appears on the website.

    unitCode: str
        Unit of the value field, one of: SQMT (square meters), SQFT (square feet), ACRE (acres).

    value: float
        Area

class zyte_common_items.Request(**kwargs)
    Describe a web request to load a page

    cast(cls: Type[RequestT]) → RequestT
        Convert value, an instance of Request or a subclass, into cls, a different class that is also either Request or a subclass.

    to_scrapy(callback, **kwargs)
        Convert a request to scrapy.Request. All kwargs are passed to scrapy.Request as-is.

    body: Optional[str]
        HTTP request body, Base64-encoded

    property body_bytes: Optional[bytes]
        Request.body as bytes

    headers: Optional[List[Header]]
        HTTP headers

    method: str
        HTTP method

    name: Optional[str]
        Name of the page being requested.

```

url: `str`
HTTP URL

class `zyte_common_items.SocialMediaPostAuthor(**kwargs)`

Details of the author of a social media post.

dateAccountCreated: `Optional[str]`

The date of the creation of the author's account.

isVerified: `Optional[bool]`

Indication if the author's account is verified.

location: `Optional[str]`

The location of the author, if it's available in the author profile. Country or city location only.

numberOfFollowers: `Optional[int]`

The number of the followers that observe the author.

numberOfFollowing: `Optional[int]`

The number of the users that the author follows.

class `zyte_common_items.StarRating(**kwargs)`

Official star rating of a place.

ratingValue: `Optional[float]`

Star rating value of the place.

raw: `Optional[str]`

Star rating of the place, as it appears on the page, without processing.

class `zyte_common_items.Url(**kwargs)`

A URL.

class `zyte_common_items.Video(**kwargs)`

Video.

See [Article.videos](#).

url: `str`

URL.

When multiple URLs exist for a given media element, pointing to different-quality versions, the highest-quality URL should be used.

[Data URIs](#) are not allowed in this attribute.

6.5.1 Item metadata components

class `zyte_common_items.Metadata(**kwargs)`

Bases: [DetailsMetadata](#)

Generic metadata class.

It defines all attributes of metadata classes for specific item types, so that it can be used during extraction instead of a more specific class, and later converted to the corresponding, more specific metadata class.

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

probability: `Optional[float]`

The probability (0 for 0%, 1 for 100%) that the resource features the expected data type.

For example, if the extraction of a product from a given URL is requested, and that URL points to the webpage of a product with complete certainty, the value should be *1*. If with complete certainty the webpage features a job listing instead of a product, the value should be *0*. When there is no complete certainty, the value could be anything in between (e.g. *0.96*).

searchText: `Optional[str]`

The search text used to find the item.

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

class `zyte_common_items.components.metadata.ProbabilityMetadata(**kwargs)`

Bases: `BaseMetadata`

Data extraction process metadata.

probability: `Optional[float]`

The probability (0 for 0%, 1 for 100%) that the resource features the expected data type.

For example, if the extraction of a product from a given URL is requested, and that URL points to the webpage of a product with complete certainty, the value should be *1*. If with complete certainty the webpage features a job listing instead of a product, the value should be *0*. When there is no complete certainty, the value could be anything in between (e.g. *0.96*).

class `zyte_common_items.components.metadata.ListMetadata(**kwargs)`

Bases: `BaseMetadata`

Minimal metadata for list item classes, such as `ProductList` or `ArticleList`.

See `ArticleList.metadata`.

get_date_downloaded_parsed() → `Optional[datetime]`

Return `dateDownloaded` as a TZ-aware datetime object

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

class `zyte_common_items.components.metadata.DetailsMetadata(**kwargs)`

Bases: `ListMetadata`

Minimal metadata for details item classes, such as `Product` or `Article`.

get_date_downloaded_parsed() → `Optional[datetime]`

Return `dateDownloaded` as a TZ-aware datetime object

dateDownloaded: `Optional[str]`

Date and time when the product data was downloaded, in UTC timezone and the following format: YYYY-MM-DDThh:mm:ssZ.

probability: `Optional[float]`

The probability (0 for 0%, 1 for 100%) that the resource features the expected data type.

For example, if the extraction of a product from a given URL is requested, and that URL points to the webpage of a product with complete certainty, the value should be *1*. If with complete certainty the webpage features a job listing instead of a product, the value should be *0*. When there is no complete certainty, the value could be anything in between (e.g. *0.96*).

validationMessages: `Optional[Dict[str, List[str]]]`

Contains paths to fields with the description of issues found with their values.

class `zyte_common_items.components.metadata.BaseMetadata(**kwargs)`

Bases: `Item`

Base metadata class

cast(*cls*: `Type[MetadataT]`) \rightarrow `MetadataT`

Convert *value*, a metadata instance, into a different metadata *cls*.

6.5.2 Typing

class `zyte_common_items.components.metadata.MetadataT`

`TypeVar` for `BaseMetadata`.

class `zyte_common_items.components.request.RequestT`

`TypeVar` for `Request`.

6.6 Converters

A module with common attrs converters

class `zyte_common_items.converters.MetadataCaster(target)`

attrs converter that converts an input metadata object into the metadata class declared by the container page object class.

`zyte_common_items.converters.to_probability_request_list(request_list)`

attrs converter to turn lists of `Request` instances into lists of `ProbabilityRequest` instances.

`zyte_common_items.converters.to_probability_request_list_optional(request_list)`

attrs converter to turn lists of `Request` instances into lists of `ProbabilityRequest` instances. If `None` is passed, `None` is returned.

`zyte_common_items.converters.url_to_str(url: Union[str, _Url]) \rightarrow str`

Return the input `RequestUrl` or `ResponseUrl` object as a string.

`zyte_common_items.converters.url_to_str_optional(url: Optional[Union[str, _Url]]) \rightarrow Optional[str]`

Return the input `RequestUrl` or `ResponseUrl` object as a string, or `None` if url is `None`.

6.7 Adapter

class `zyte_common_items.ZyteItemAdapter(item: Any)`

Wrap an *item* to interact with its content as if it was a dictionary.

It can be *configured* into *itemadapter* to improve interaction with *items* for itemadapter users like *Scrapy*.

In extends *AttrsAdapter* with the following features:

- Allows interaction and serialization of fields from *_unknown_fields_dict* as if they were regular item fields.
- Removes keys with empty values from the output of *ItemAdapter.asdict()*, for a cleaner output.

class `zyte_common_items.ZyteItemKeepEmptyAdapter(item: Any)`

Similar to *ZyteItemAdapter* but doesn't remove empty values.

It is intended to be used in tests and other use cases where it's important to differentiate between empty and missing fields.

6.8 Scrapy Pipelines

class `zyte_common_items.pipelines.AEPipeline`

Replace standard items with matching items with the old Zyte Automatic Extraction schema.

This item pipeline is intended to help in the *migration from Zyte Automatic Extraction to Zyte API automatic extraction*.

In the simplest scenarios, it can be added to the `ITEM_PIPELINES` setting in migrated code to ensure that the schema of output items matches the old schema.

In scenarios where page object classes were being used to fix, extend or customize extraction, it is recommended to migrate page object classes to the new schemas, or move page object class code to the corresponding spider callback.

If you have callbacks with custom code based on the old schema, you can either migrate that code, and ideally move it to a page object class, or use `zyte_common_items.ae.downgrade` at the beginning of the callback, e.g.:

```
from zyte_common_items import ae

...

def parse_product(self, response: DummyResponse, product: Product):
    product = ae.downgrade(product)

...
```

class `zyte_common_items.pipelines.DropLowProbabilityItemPipeline(crawler)`

This pipeline drops an item if its probability, defined in the settings, is less than the specified threshold.

By default, 0.1 threshold is used, i.e. items with probability < 0.1 are dropped.

You can customize the thresholds by using the `ITEM_PROBABILITY_THRESHOLDS` setting that offers greater flexibility, allowing you to define thresholds for each Item class separately or set a default threshold for all other item classes.

Thresholds for Item classes can be defined using either the path to the Item class or directly using the Item classes themselves.

The example of using ITEM_PROBABILITY_THRESHOLDS:

```
from zyte_common_items import Article

ITEM_PROBABILITY_THRESHOLDS = {
    Article: 0.2,
    "zyte_common_items.Product": 0.3,
    "default": 0.15,
}
```

CHANGELOG

7.1 0.19.0 (2024-04-24)

- Now requires `attrs >= 22.2.0`.
- New deprecations:
 - `zyte_common_items.components.request_list_processor` (use `zyte_common_items.processors.probability_request_list_processor`)
 - `zyte_common_items.items.RequestListCaster` (use `zyte_common_items.converters.to_probability_request_list`)
 - `zyte_common_items.util.metadata_processor` (use `zyte_common_items.processors.metadata_processor`)
- Added *[DropLowProbabilityItemPipeline](#)* that drops items with the probability value lower than a set threshold.
- Added the *[BaseMetadata](#)*, *[ListMetadata](#)*, and *[DetailMetadata](#)* classes (they were previously private).
- Added the *[ListMetadata.validationMessages](#)* attribute.
- Added the *[ListMetadata.get_date_downloaded_parsed\(\)](#)* method.
- Added the *[zyte_common_items.converters](#)* module with useful `attrs` converters.
- Reorganized the module structure.
- Documentation improvements.
- Test and CI fixes and improvements.

7.2 0.18.0 (2024-03-15)

- Initial support for *[request templates](#)*, starting with search requests.

7.3 0.17.1 (2024-03-13)

- Added Python 3.12 support.
- `description_processor()` and `description_html_processor()` now raise an exception when they receive an unsupported input value such as a non-`HtmlElement` node.
- Documentation improvements.

7.4 0.17.0 (2024-02-14)

- Implement the `zyte_common_items.ae` module and the `zyte_common_items.pipelines.AEPipeline` item pipeline to make it easier to migrate from Zyte Automatic Extraction to Zyte API automatic extraction.

7.5 0.16.0 (2024-02-06)

- Auto-prefixed versions of *page objects*, such as `AutoProductPage()`, now have all their fields defined as synchronous instead of asynchronous.

7.6 0.15.0 (2024-01-30)

- Now requires `zyte-parsers >= 0.5.0`.
- Added `SocialMediaPost` and related classes.
- Added `ProductFromListExtractor`, `ProductFromListSelectorExtractor`, `ProductVariantExtractor` and `ProductVariantSelectorExtractor`.
- Added `zyte_common_items.processors.rating_processor()` and enabled it for the `aggregateRating` fields in the page classes for `BusinessPlace` and `Product`.
- Improved the documentation about the processors.

7.7 0.14.0 (2024-01-16)

- Now requires `zyte-parsers >= 0.4.0`.
- Added `zyte_common_items.processors.gtin_processor()` and enabled it for the `gtin` fields in the page classes for `Product`.
- Improved the API documentation.

7.8 0.13.0 (2023-11-09)

- Added Auto-prefixed versions of *page objects*, such as `AutoProductPage()`, that return data from Zyte API automatic extraction from their fields by default, and can be used to more easily override that data with custom parsing logic.

7.9 0.12.0 (2023-10-27)

- Added `get_probability()` helper method in item classes (e.g. `Product`, `Article`) and `ProbabilityRequest`.

7.10 0.11.0 (2023-09-08)

- Now requires `clear-html` $\geq 0.4.0$.
- Added `zyte_common_items.processors.description_processor()` and enabled it for the description fields in the page classes for `BusinessPlace`, `JobPosting`, `Product` and `RealEstate`.
- Added `zyte_common_items.processors.description_html_processor()` and enabled it for the descriptionHtml fields in the page classes for `JobPosting` and `Product`.
- Added default implementations for the description (in the page classes for `BusinessPlace`, `JobPosting`, `Product` and `RealEstate`) and descriptionHtml (in the page classes for `JobPosting` and `Product`) fields: if one of these fields is user-defined, another one will use it.
- `price_processor()` and `simple_price_processor()` now keep at least two decimal places when formatting the result.

7.11 0.10.0 (2023-08-24)

- Now requires `price-parser` $\geq 0.3.4$ (a new dependency) and `zyte-parsers` $\geq 0.3.0$ (a version increase).
- Added `zyte_common_items.processors.price_processor()` and enabled it for the price fields.
- Added `zyte_common_items.processors.simple_price_processor()` and enabled it for the regularPrice fields.
- Added default implementations for the currency (uses the CURRENCY attribute on the page class) and currencyRaw (uses the data extracted by the price field) fields.

7.12 0.9.0 (2023-08-03)

- Now requires `web-poet` $\geq 0.14.0$.
- Fixed detection of the `HasMetadata` base class.

7.13 0.8.0 (2023-07-27)

- Updated minimum versions for the following requirements:
 - `attrs` `>= 22.1.0`
 - `web-poet` `>= 0.9.0`
 - `zyte-parsers` `>= 0.2.0`
- Added `JobPosting` and related classes.
- Added `zyte_common_items.processors.brand_processor()` and enabled it for the brand fields.
- Added `zyte_common_items.Request.to_scrapy()` to convert `zyte_common_items.Request` instances to `scrapy.http.Request` instances.

7.14 0.7.0 (2023-07-11)

- Now requires `zyte-parsers`.
- Added navigation classes: `ArticleNavigation`, `ProductNavigation`, the page classes that produce them, and other related classes.
- Improved the metadata field handling, also fixing some bugs:
 - Added *item-specific metadata classes*. The metadata item fields were changed to use them.
 - **Backwards incompatible change:** the `DateDownloadedMetadata` class was removed. The item-specific ones are now used instead.
 - **Backwards incompatible change:** `ArticleFromList` no longer has a `probability` field and instead has a `metadata` field like all other similar classes.
 - **Backwards incompatible change:** while in most items the old and the new type of the metadata field have the same fields, the one in `Article` now has `probability`, the one in `ProductList` no longer has `probability`, and the one in `ProductFromList` no longer has `dateDownloaded`.
 - The default `probability` value is now `1.0` instead of `None`.
 - Added the `HasMetadata` mixin which is used similarly to `Returns` to set the page metadata class.
 - Metadata objects assigned to the `metadata` fields of the items or returned from the `metadata()` methods of the pages are now converted to suitable classes.
- Added `zyte_common_items.processors.breadcrumbs_processor()` and enabled it for the breadcrumbs fields.

7.15 0.6.0 (2023-07-05)

- Added `Article` and `ArticleList`.
- Added support for Python 3.11 and dropped support for Python 3.7.

7.16 0.5.0 (2023-05-10)

- Now requires `itemadapter >= 0.8.0`.
- Added `RealEstate`.
- Added the `zyte_common_items.BasePage.no_item_found()` and `zyte_common_items.Page.no_item_found()` methods.
- Improved the error message for invalid input.
- Added `ZyteItemKeepEmptyAdapter` and documented how to use it and `ZyteItemAdapter` in custom subclasses of `itemadapter.ItemAdapter`.

7.17 0.4.0 (2023-03-27)

- Added support for business places.

7.18 0.3.1 (2023-03-17)

- Fixed fields from `BasePage` subclasses leaking across subclasses. (#29, #30)
- Improved how the `from_dict()` and `from_list()` methods report issues in the input data. (#25)

7.19 0.3.0 (2023-02-03)

- Added *page object classes* for e-commerce product detail and product list pages.

7.20 0.2.0 (2022-09-22)

- Supports `web_poet.RequestUrl` and `web_poet.ResponseUrl` and automatically convert them into a string on URL fields like `Product.url`.
- Bumps the `web_poet` dependency version from `0.4.0` to `0.5.0` which fully supports type hints using the `py.typed` marker.
- This package now also supports type hints using the `py.typed` marker. This means mypy would properly use the type annotations in the items when using it in your project.
- Minor improvements in tests and annotations.

7.21 0.1.0 (2022-07-29)

Initial release.

CONTRIBUTING

You can contribute to this project with code.

To prepare your development environment:

1. Clone the [source code](#):

```
git clone https://github.com/zytedata/zyte-common-items.git
cd zyte-common-items
```

2. Create and activate a [Python virtual environment](#):

```
python -m venv venv
. venv/bin/activate
```

3. Install the packages needed for development:

```
pip install -r requirements-dev.txt
```

4. Configure our [Git pre-commit hooks](#):

```
pre-commit install
```

You can search our [issue tracker](#) for pending work, and start a pull request for any pending issue that is not actively being worked on already, no need to ask for permission first.

If there is something else you wish to implement, please open an issue first to open a discussion about it, before you work on a pull request. You probably do not want to spend time on a pull request to later be told that the feature does not fit the project plans in the first place.

PYTHON MODULE INDEX

Z

`zyte_common_items.converters`, [54](#)

Symbols

`_BasePage` (class in `zyte_common_items.pages.base`), 43
`_unknown_fields_dict` (`zyte_common_items.Item` attribute), 39

A

`actions` (`zyte_common_items.BusinessPlace` attribute), 31
`additionalProperties` (`zyte_common_items.BusinessPlace` attribute), 31
`additionalProperties` (`zyte_common_items.Product` attribute), 17
`additionalProperties` (`zyte_common_items.ProductVariant` attribute), 20
`additionalProperties` (`zyte_common_items.RealEstate` attribute), 33
`AdditionalProperty` (class in `zyte_common_items`), 46
`Address` (class in `zyte_common_items`), 46
`address` (`zyte_common_items.BusinessPlace` attribute), 31
`address` (`zyte_common_items.RealEstate` attribute), 33
`addressCity` (`zyte_common_items.Address` attribute), 46
`addressCountry` (`zyte_common_items.Address` attribute), 46
`addressLocality` (`zyte_common_items.Address` attribute), 47
`addressRaw` (`zyte_common_items.Address` attribute), 47
`addressRegion` (`zyte_common_items.Address` attribute), 47
`AEPipeline` (class in `zyte_common_items.pipelines`), 55
`AggregateRating` (class in `zyte_common_items`), 47
`aggregateRating` (`zyte_common_items.BusinessPlace` attribute), 31
`aggregateRating` (`zyte_common_items.Product` attribute), 17
`Amenity` (class in `zyte_common_items`), 47
`amenityFeatures` (`zyte_common_items.BusinessPlace` attribute), 31

`area` (`zyte_common_items.RealEstate` attribute), 33
`areaType` (`zyte_common_items.RealEstateArea` attribute), 51
`Article` (class in `zyte_common_items`), 26
`articleBody` (`zyte_common_items.Article` attribute), 26
`articleBody` (`zyte_common_items.ArticleFromList` attribute), 28
`articleBodyHtml` (`zyte_common_items.Article` attribute), 26
`ArticleFromList` (class in `zyte_common_items`), 28
`ArticleList` (class in `zyte_common_items`), 28
`ArticleListMetadata` (class in `zyte_common_items`), 29
`ArticleListPage` (class in `zyte_common_items`), 41
`ArticleMetadata` (class in `zyte_common_items`), 27
`ArticleNavigation` (class in `zyte_common_items`), 30
`ArticleNavigationMetadata` (class in `zyte_common_items`), 30
`ArticleNavigationPage` (class in `zyte_common_items`), 41
`ArticlePage` (class in `zyte_common_items`), 41
`articles` (`zyte_common_items.ArticleList` attribute), 28
`Audio` (class in `zyte_common_items`), 47
`audios` (`zyte_common_items.Article` attribute), 26
`Author` (class in `zyte_common_items`), 48
`author` (`zyte_common_items.SocialMediaPost` attribute), 37
`authors` (`zyte_common_items.Article` attribute), 26
`authors` (`zyte_common_items.ArticleFromList` attribute), 29
`AutoArticleListPage` (class in `zyte_common_items`), 41
`AutoArticleNavigationPage` (class in `zyte_common_items`), 41
`AutoArticlePage` (class in `zyte_common_items`), 41
`AutoBusinessPlacePage` (class in `zyte_common_items`), 42
`AutoJobPostingPage` (class in `zyte_common_items`), 42
`AutoProductListPage` (class in `zyte_common_items`), 40
`AutoProductNavigationPage` (class in `zyte_common_items`), 40

AutoProductPage (class in *zyte_common_items*), 40
AutoRealEstatePage (class in *zyte_common_items*), 42
AutoSocialMediaPostPage (class in *zyte_common_items*), 42
availability (*zyte_common_items.Product* attribute), 17
availability (*zyte_common_items.ProductVariant* attribute), 20

B

BaseArticleListPage (class in *zyte_common_items*), 41
BaseArticleNavigationPage (class in *zyte_common_items*), 41
BaseArticlePage (class in *zyte_common_items*), 41
BaseBusinessPlacePage (class in *zyte_common_items*), 41
BaseJobPostingPage (class in *zyte_common_items*), 42
BaseMetadata (class in *zyte_common_items.components.metadata*), 54
BasePage (class in *zyte_common_items*), 43
BasePage.metadata (in module *zyte_common_items*), 43
BasePage.url (in module *zyte_common_items*), 43
BaseProductListPage (class in *zyte_common_items*), 40
BaseProductNavigationPage (class in *zyte_common_items*), 40
BaseProductPage (class in *zyte_common_items*), 40
BaseRealEstatePage (class in *zyte_common_items*), 42
BaseSalary (class in *zyte_common_items*), 48
baseSalary (*zyte_common_items.JobPosting* attribute), 35
BaseSocialMediaPostPage (class in *zyte_common_items*), 42
bestRating (*zyte_common_items.AggregateRating* attribute), 47
body (*zyte_common_items.Request* attribute), 51
body (*zyte_common_items.SearchRequestTemplate* attribute), 38
body_bytes (*zyte_common_items.Request* property), 51
Brand (class in *zyte_common_items*), 48
brand (*zyte_common_items.Product* attribute), 17
brand_processor() (in module *zyte_common_items.processors*), 45
Breadcrumb (class in *zyte_common_items*), 48
breadcrumbs (*zyte_common_items.Article* attribute), 26
breadcrumbs (*zyte_common_items.ArticleList* attribute), 28
breadcrumbs (*zyte_common_items.Product* attribute), 17
breadcrumbs (*zyte_common_items.ProductList* attribute), 23

breadcrumbs (*zyte_common_items.RealEstate* attribute), 33
breadcrumbs_processor() (in module *zyte_common_items.processors*), 45
BusinessPlace (class in *zyte_common_items*), 31
BusinessPlaceMetadata (class in *zyte_common_items*), 32
BusinessPlacePage (class in *zyte_common_items*), 41

C

canonicalUrl (*zyte_common_items.Article* attribute), 26
canonicalUrl (*zyte_common_items.ArticleList* attribute), 28
canonicalUrl (*zyte_common_items.Product* attribute), 17
canonicalUrl (*zyte_common_items.ProductList* attribute), 23
canonicalUrl (*zyte_common_items.ProductVariant* attribute), 21
cast() (*zyte_common_items.components.metadata.BaseMetadata* method), 54
cast() (*zyte_common_items.Request* method), 51
categories (*zyte_common_items.BusinessPlace* attribute), 31
categoryName (*zyte_common_items.ArticleNavigation* attribute), 30
categoryName (*zyte_common_items.ProductList* attribute), 23
categoryName (*zyte_common_items.ProductNavigation* attribute), 25
closes (*zyte_common_items.OpeningHoursItem* attribute), 50
color (*zyte_common_items.Product* attribute), 18
color (*zyte_common_items.ProductVariant* attribute), 21
containedInPlace (*zyte_common_items.BusinessPlace* attribute), 31
currency (*zyte_common_items.BaseSalary* attribute), 48
currency (*zyte_common_items.Product* attribute), 18
currency (*zyte_common_items.ProductFromList* attribute), 24
currency (*zyte_common_items.ProductVariant* attribute), 21
currency (*zyte_common_items.RealEstate* attribute), 33
currencyRaw (*zyte_common_items.BaseSalary* attribute), 48
currencyRaw (*zyte_common_items.Product* attribute), 18
currencyRaw (*zyte_common_items.ProductFromList* attribute), 24
currencyRaw (*zyte_common_items.ProductVariant* attribute), 21
currencyRaw (*zyte_common_items.RealEstate* attribute), 33

D

[dateAccountCreated](#) ([zyte_common_items.SocialMediaPost](#) attribute), 52
[dateDownloaded](#) ([zyte_common_items.ArticleListMetadata](#) attribute), 29
[dateDownloaded](#) ([zyte_common_items.ArticleMetadata](#) attribute), 27
[dateDownloaded](#) ([zyte_common_items.ArticleNavigationMetadata](#) attribute), 30
[dateDownloaded](#) ([zyte_common_items.BusinessPlaceMetadata](#) attribute), 32
[dateDownloaded](#) ([zyte_common_items.components.metadata.DetailsMetadata](#) attribute), 53
[dateDownloaded](#) ([zyte_common_items.components.metadata.ListMetadata](#) attribute), 53
[dateDownloaded](#) ([zyte_common_items.JobPostingMetadata](#) attribute), 37
[dateDownloaded](#) ([zyte_common_items.Metadata](#) attribute), 52
[dateDownloaded](#) ([zyte_common_items.ProductListMetadata](#) attribute), 24
[dateDownloaded](#) ([zyte_common_items.ProductMetadata](#) attribute), 22
[dateDownloaded](#) ([zyte_common_items.ProductNavigationMetadata](#) attribute), 25
[dateDownloaded](#) ([zyte_common_items.RealEstateMetadata](#) attribute), 34
[dateDownloaded](#) ([zyte_common_items.SearchRequestTemplateMetadata](#) attribute), 39
[dateDownloaded](#) ([zyte_common_items.SocialMediaPostMetadata](#) attribute), 38
[dateModified](#) ([zyte_common_items.Article](#) attribute), 26
[dateModified](#) ([zyte_common_items.JobPosting](#) attribute), 35
[dateModifiedRaw](#) ([zyte_common_items.Article](#) attribute), 26
[dateModifiedRaw](#) ([zyte_common_items.JobPosting](#) attribute), 35
[datePublished](#) ([zyte_common_items.Article](#) attribute), 26
[datePublished](#) ([zyte_common_items.ArticleFromList](#) attribute), 29
[datePublished](#) ([zyte_common_items.JobPosting](#) attribute), 35
[datePublished](#) ([zyte_common_items.RealEstate](#) attribute), 33
[datePublished](#) ([zyte_common_items.SocialMediaPost](#) attribute), 37
[datePublishedRaw](#) ([zyte_common_items.Article](#) attribute), 27
[datePublishedRaw](#) ([zyte_common_items.ArticleFromList](#) attribute), 29
[datePublishedRaw](#) ([zyte_common_items.JobPosting](#) attribute), 35
[datePublishedRaw](#) ([zyte_common_items.RealEstate](#) attribute), 33
[dayOfWeek](#) ([zyte_common_items.OpeningHoursItem](#) attribute), 50
[description](#) ([zyte_common_items.Article](#) attribute), 27
[description](#) ([zyte_common_items.BusinessPlace](#) attribute), 31
[description](#) ([zyte_common_items.JobPosting](#) attribute), 35
[description](#) ([zyte_common_items.Product](#) attribute), 18
[description](#) ([zyte_common_items.RealEstate](#) attribute), 33
[description_html_processor\(\)](#) (in module [zyte_common_items.processors](#)), 45
[description_processor\(\)](#) (in module [zyte_common_items.processors](#)), 45
[descriptionHtml](#) ([zyte_common_items.JobPosting](#) attribute), 36
[descriptionHtml](#) ([zyte_common_items.Product](#) attribute), 18
[DescriptionMixin](#) (class in module [zyte_common_items.pages](#)), 43
[DescriptionMixin.description](#) (in module [zyte_common_items.pages](#)), 43
[DescriptionMixin.descriptionHtml](#) (in module [zyte_common_items.pages](#)), 43
[DetailsMetadata](#) (class in module [zyte_common_items.components.metadata](#)), 53
[dislikes](#) ([zyte_common_items.Reactions](#) attribute), 51
[DropLowProbabilityItemPipeline](#) (class in module [zyte_common_items.pipelines](#)), 55

E

[email](#) ([zyte_common_items.Author](#) attribute), 48
[employmentType](#) ([zyte_common_items.JobPosting](#) attribute), 36

F

[features](#) ([zyte_common_items.BusinessPlace](#) attribute), 31
[features](#) ([zyte_common_items.Product](#) attribute), 18
[from_dict\(\)](#) ([zyte_common_items.Article](#) class method), 26
[from_dict\(\)](#) ([zyte_common_items.ArticleFromList](#) class method), 28
[from_dict\(\)](#) ([zyte_common_items.ArticleList](#) class method), 28
[from_dict\(\)](#) ([zyte_common_items.ArticleNavigation](#) class method), 30
[from_dict\(\)](#) ([zyte_common_items.BusinessPlace](#) class method), 31

[from_dict\(\)](#) (*zyte_common_items.Item* class method), 39
[from_dict\(\)](#) (*zyte_common_items.JobPosting* class method), 35
[from_dict\(\)](#) (*zyte_common_items.Product* class method), 17
[from_dict\(\)](#) (*zyte_common_items.ProductFromList* class method), 23
[from_dict\(\)](#) (*zyte_common_items.ProductList* class method), 23
[from_dict\(\)](#) (*zyte_common_items.ProductNavigation* class method), 25
[from_dict\(\)](#) (*zyte_common_items.ProductVariant* class method), 20
[from_dict\(\)](#) (*zyte_common_items.RealEstate* class method), 33
[from_dict\(\)](#) (*zyte_common_items.SearchRequestTemplate* class method), 38
[from_dict\(\)](#) (*zyte_common_items.SocialMediaPost* class method), 37
[from_list\(\)](#) (*zyte_common_items.Article* class method), 26
[from_list\(\)](#) (*zyte_common_items.ArticleFromList* class method), 28
[from_list\(\)](#) (*zyte_common_items.ArticleList* class method), 28
[from_list\(\)](#) (*zyte_common_items.ArticleNavigation* class method), 30
[from_list\(\)](#) (*zyte_common_items.BusinessPlace* class method), 31
[from_list\(\)](#) (*zyte_common_items.Item* class method), 39
[from_list\(\)](#) (*zyte_common_items.JobPosting* class method), 35
[from_list\(\)](#) (*zyte_common_items.Product* class method), 17
[from_list\(\)](#) (*zyte_common_items.ProductFromList* class method), 24
[from_list\(\)](#) (*zyte_common_items.ProductList* class method), 23
[from_list\(\)](#) (*zyte_common_items.ProductNavigation* class method), 25
[from_list\(\)](#) (*zyte_common_items.ProductVariant* class method), 20
[from_list\(\)](#) (*zyte_common_items.RealEstate* class method), 33
[from_list\(\)](#) (*zyte_common_items.SearchRequestTemplate* class method), 38
[from_list\(\)](#) (*zyte_common_items.SocialMediaPost* class method), 37

G

[get_date_downloaded_parsed\(\)](#) (*zyte_common_items.components.metadata.DetailMetadata* method), 53
[get_date_downloaded_parsed\(\)](#) (*zyte_common_items.components.metadata.ListMetadata* method), 53
[get_probability\(\)](#) (*zyte_common_items.Article* method), 26
[get_probability\(\)](#) (*zyte_common_items.ArticleFromList* method), 28
[get_probability\(\)](#) (*zyte_common_items.ArticleList* method), 28
[get_probability\(\)](#) (*zyte_common_items.ArticleNavigation* method), 30
[get_probability\(\)](#) (*zyte_common_items.base.ProbabilityMixin* method), 39
[get_probability\(\)](#) (*zyte_common_items.BusinessPlace* method), 31
[get_probability\(\)](#) (*zyte_common_items.Item* method), 39
[get_probability\(\)](#) (*zyte_common_items.JobPosting* method), 35
[get_probability\(\)](#) (*zyte_common_items.Product* method), 17
[get_probability\(\)](#) (*zyte_common_items.ProductFromList* method), 24
[get_probability\(\)](#) (*zyte_common_items.ProductList* method), 23
[get_probability\(\)](#) (*zyte_common_items.ProductNavigation* method), 25
[get_probability\(\)](#) (*zyte_common_items.ProductVariant* method), 20
[get_probability\(\)](#) (*zyte_common_items.RealEstate* method), 33
[get_probability\(\)](#) (*zyte_common_items.SearchRequestTemplate* method), 38
[get_probability\(\)](#) (*zyte_common_items.SocialMediaPost* method), 37
[Gtin](#) (class in *zyte_common_items*), 49
[gtin](#) (*zyte_common_items.Product* attribute), 18
[gtin](#) (*zyte_common_items.ProductVariant* attribute), 21
[gtin_processor\(\)](#) (in *zyte_common_items.processors* module), 45

H

[hashtags](#) (*zyte_common_items.SocialMediaPost* attribute), 37
[HasMetadata](#) (class in *zyte_common_items*), 44
[Header](#) (class in *zyte_common_items*), 49
[headers](#) (*zyte_common_items.Request* attribute), 51
[headers](#) (*zyte_common_items.SearchRequestTemplate* attribute), 38
[headline](#) (*zyte_common_items.Article* attribute), 27
[headline](#) (*zyte_common_items.ArticleFromList* attribute), 29
[headline](#) (*zyte_common_items.JobPosting* attribute), 36

HiringOrganization (class in `zyte_common_items`), 49
 hiringOrganization (`zyte_common_items.JobPosting` attribute), 36

I

id (`zyte_common_items.HiringOrganization` attribute), 49
 Image (class in `zyte_common_items`), 49
 images (`zyte_common_items.Article` attribute), 27
 images (`zyte_common_items.ArticleFromList` attribute), 29
 images (`zyte_common_items.BusinessPlace` attribute), 31
 images (`zyte_common_items.Product` attribute), 18
 images (`zyte_common_items.ProductVariant` attribute), 21
 images (`zyte_common_items.RealEstate` attribute), 34
 inLanguage (`zyte_common_items.Article` attribute), 27
 inLanguage (`zyte_common_items.ArticleFromList` attribute), 29
 isVerified (`zyte_common_items.BusinessPlace` attribute), 31
 isVerified (`zyte_common_items.SocialMediaPostAuthor` attribute), 52
 Item (class in `zyte_common_items`), 39
 items (`zyte_common_items.ArticleNavigation` attribute), 30
 items (`zyte_common_items.ProductNavigation` attribute), 25

J

JobLocation (class in `zyte_common_items`), 49
 jobLocation (`zyte_common_items.JobPosting` attribute), 36
 JobPosting (class in `zyte_common_items`), 35
 jobPostingId (`zyte_common_items.JobPosting` attribute), 36
 JobPostingMetadata (class in `zyte_common_items`), 36
 JobPostingPage (class in `zyte_common_items`), 42
 jobStartDate (`zyte_common_items.JobPosting` attribute), 36
 jobStartDateRaw (`zyte_common_items.JobPosting` attribute), 36
 jobTitle (`zyte_common_items.JobPosting` attribute), 36

L

latitude (`zyte_common_items.Address` attribute), 47
 likes (`zyte_common_items.Reactions` attribute), 51
 Link (class in `zyte_common_items`), 49
 ListMetadata (class in `zyte_common_items.components.metadata`), 53
 location (`zyte_common_items.SocialMediaPostAuthor` attribute), 52
 longitude (`zyte_common_items.Address` attribute), 47

M

mainImage (`zyte_common_items.Article` attribute), 27
 mainImage (`zyte_common_items.ArticleFromList` attribute), 29
 mainImage (`zyte_common_items.Product` attribute), 19
 mainImage (`zyte_common_items.ProductFromList` attribute), 24
 mainImage (`zyte_common_items.ProductVariant` attribute), 21
 mainImage (`zyte_common_items.RealEstate` attribute), 34
 map (`zyte_common_items.BusinessPlace` attribute), 31
 mediaUrls (`zyte_common_items.SocialMediaPost` attribute), 37
 Metadata (class in `zyte_common_items`), 52
 metadata (`zyte_common_items.Article` attribute), 27
 metadata (`zyte_common_items.ArticleFromList` attribute), 29
 metadata (`zyte_common_items.ArticleList` attribute), 28
 metadata (`zyte_common_items.ArticleNavigation` attribute), 30
 metadata (`zyte_common_items.BusinessPlace` attribute), 31
 metadata (`zyte_common_items.JobPosting` attribute), 36
 metadata (`zyte_common_items.ProbabilityRequest` attribute), 51
 metadata (`zyte_common_items.Product` attribute), 19
 metadata (`zyte_common_items.ProductFromList` attribute), 24
 metadata (`zyte_common_items.ProductList` attribute), 23
 metadata (`zyte_common_items.ProductNavigation` attribute), 25
 metadata (`zyte_common_items.RealEstate` attribute), 34
 metadata (`zyte_common_items.SearchRequestTemplate` attribute), 38
 metadata (`zyte_common_items.SocialMediaPost` attribute), 37
 MetadataCaster (class in `zyte_common_items.converters`), 54
 MetadataT (class in `zyte_common_items.components.metadata`), 54
 method (`zyte_common_items.Request` attribute), 51
 method (`zyte_common_items.SearchRequestTemplate` attribute), 39
 module
 `zyte_common_items.converters`, 54
 mpn (`zyte_common_items.Product` attribute), 19
 mpn (`zyte_common_items.ProductVariant` attribute), 21

N

name (`zyte_common_items.AdditionalProperty` attribute), 46
 name (`zyte_common_items.Amenity` attribute), 47

`name` (*zyte_common_items.Author* attribute), 48
`name` (*zyte_common_items.Brand* attribute), 48
`name` (*zyte_common_items.Breadcrumb* attribute), 48
`name` (*zyte_common_items.BusinessPlace* attribute), 31
`name` (*zyte_common_items.Header* attribute), 49
`name` (*zyte_common_items.HiringOrganization* attribute), 49
`name` (*zyte_common_items.NamedLink* attribute), 50
`name` (*zyte_common_items.ParentPlace* attribute), 50
`name` (*zyte_common_items.Product* attribute), 19
`name` (*zyte_common_items.ProductFromList* attribute), 24
`name` (*zyte_common_items.ProductVariant* attribute), 21
`name` (*zyte_common_items.RealEstate* attribute), 34
`name` (*zyte_common_items.Request* attribute), 51
`NamedLink` (class in *zyte_common_items*), 50
`nameRaw` (*zyte_common_items.Author* attribute), 48
`nameRaw` (*zyte_common_items.HiringOrganization* attribute), 49
`nextPage` (*zyte_common_items.ArticleNavigation* attribute), 30
`nextPage` (*zyte_common_items.ProductNavigation* attribute), 25
`no_item_found()` (*zyte_common_items.BasePage* method), 43
`no_item_found()` (*zyte_common_items.Page* method), 44
`numberOfBathroomsTotal` (*zyte_common_items.RealEstate* attribute), 34
`numberOfBedrooms` (*zyte_common_items.RealEstate* attribute), 34
`numberOfFollowers` (*zyte_common_items.SocialMediaPostAuthor* attribute), 52
`numberOfFollowing` (*zyte_common_items.SocialMediaPostAuthor* attribute), 52
`numberOfFullBathrooms` (*zyte_common_items.RealEstate* attribute), 34
`numberOfPartialBathrooms` (*zyte_common_items.RealEstate* attribute), 34
`numberOfRooms` (*zyte_common_items.RealEstate* attribute), 34

O

`openingHours` (*zyte_common_items.BusinessPlace* attribute), 31
`OpeningHoursItem` (class in *zyte_common_items*), 50
`opens` (*zyte_common_items.OpeningHoursItem* attribute), 50

P

`Page` (class in *zyte_common_items*), 44

`Page.metadata` (in module *zyte_common_items*), 44
`Page.url` (in module *zyte_common_items*), 44
`pageNumber` (*zyte_common_items.ArticleNavigation* attribute), 30
`pageNumber` (*zyte_common_items.ProductList* attribute), 23
`pageNumber` (*zyte_common_items.ProductNavigation* attribute), 25
`paginationNext` (*zyte_common_items.ProductList* attribute), 23
`ParentPlace` (class in *zyte_common_items*), 50
`placeId` (*zyte_common_items.BusinessPlace* attribute), 32
`placeId` (*zyte_common_items.ParentPlace* attribute), 50
`postalCode` (*zyte_common_items.Address* attribute), 47
`postalCodeAux` (*zyte_common_items.Address* attribute), 47
`postId` (*zyte_common_items.SocialMediaPost* attribute), 37
`price` (*zyte_common_items.Product* attribute), 19
`price` (*zyte_common_items.ProductFromList* attribute), 24
`price` (*zyte_common_items.ProductVariant* attribute), 21
`price` (*zyte_common_items.RealEstate* attribute), 34
`price_processor()` (in module *zyte_common_items.processors*), 45
`PriceMixin` (class in *zyte_common_items.pages*), 43
`PriceMixin.currency` (in module *zyte_common_items.pages*), 43
`PriceMixin.currencyRaw` (in module *zyte_common_items.pages*), 43
`priceRange` (*zyte_common_items.BusinessPlace* attribute), 32
`probability` (*zyte_common_items.ArticleMetadata* attribute), 27
`probability` (*zyte_common_items.BusinessPlaceMetadata* attribute), 32
`probability` (*zyte_common_items.components.metadata.DetailsMetadata* attribute), 53
`probability` (*zyte_common_items.components.metadata.ProbabilityMetadata* attribute), 53
`probability` (*zyte_common_items.JobPostingMetadata* attribute), 37
`probability` (*zyte_common_items.Metadata* attribute), 53
`probability` (*zyte_common_items.ProbabilityMetadata* attribute), 50
`probability` (*zyte_common_items.ProductMetadata* attribute), 22
`probability` (*zyte_common_items.RealEstateMetadata* attribute), 35
`probability` (*zyte_common_items.SearchRequestTemplateMetadata* attribute), 39
`probability` (*zyte_common_items.SocialMediaPostMetadata*

- [attribute](#)), 38
[ProbabilityMetadata](#) (class in [zyte_common_items](#)), 50
[ProbabilityMetadata](#) (class in [zyte_common_items.components.metadata](#)), 53
[ProbabilityMixin](#) (class in [zyte_common_items.base](#)), 39
[ProbabilityRequest](#) (class in [zyte_common_items](#)), 50
[Product](#) (class in [zyte_common_items](#)), 17
[ProductFromList](#) (class in [zyte_common_items](#)), 23
[ProductFromListExtractor](#) (class in [zyte_common_items](#)), 44
[ProductFromListSelectorExtractor](#) (class in [zyte_common_items](#)), 44
[productId](#) ([zyte_common_items.Product](#) attribute), 19
[productId](#) ([zyte_common_items.ProductFromList](#) attribute), 24
[productId](#) ([zyte_common_items.ProductVariant](#) attribute), 21
[ProductList](#) (class in [zyte_common_items](#)), 23
[ProductListMetadata](#) (class in [zyte_common_items](#)), 24
[ProductListPage](#) (class in [zyte_common_items](#)), 40
[ProductMetadata](#) (class in [zyte_common_items](#)), 22
[ProductNavigation](#) (class in [zyte_common_items](#)), 25
[ProductNavigationMetadata](#) (class in [zyte_common_items](#)), 25
[ProductNavigationPage](#) (class in [zyte_common_items](#)), 40
[ProductPage](#) (class in [zyte_common_items](#)), 40
[products](#) ([zyte_common_items.ProductList](#) attribute), 23
[ProductVariant](#) (class in [zyte_common_items](#)), 20
[ProductVariantExtractor](#) (class in [zyte_common_items](#)), 44
[ProductVariantSelectorExtractor](#) (class in [zyte_common_items](#)), 44
[propertyType](#) ([zyte_common_items.RealEstate](#) attribute), 34
- ## R
- [rateType](#) ([zyte_common_items.BaseSalary](#) attribute), 48
[rating_processor\(\)](#) (in module [zyte_common_items.processors](#)), 45
[ratingValue](#) ([zyte_common_items.AggregateRating](#) attribute), 47
[ratingValue](#) ([zyte_common_items.StarRating](#) attribute), 52
[raw](#) ([zyte_common_items.BaseSalary](#) attribute), 48
[raw](#) ([zyte_common_items.JobLocation](#) attribute), 49
[raw](#) ([zyte_common_items.RealEstateArea](#) attribute), 51
[raw](#) ([zyte_common_items.StarRating](#) attribute), 52
[rawCloses](#) ([zyte_common_items.OpeningHoursItem](#) attribute), 50
[rawDayOfWeek](#) ([zyte_common_items.OpeningHoursItem](#) attribute), 50
[rawOpens](#) ([zyte_common_items.OpeningHoursItem](#) attribute), 50
[Reactions](#) (class in [zyte_common_items](#)), 51
[reactions](#) ([zyte_common_items.SocialMediaPost](#) attribute), 37
[RealEstate](#) (class in [zyte_common_items](#)), 33
[RealEstateArea](#) (class in [zyte_common_items](#)), 51
[realEstateId](#) ([zyte_common_items.RealEstate](#) attribute), 34
[RealEstateMetadata](#) (class in [zyte_common_items](#)), 34
[RealEstatePage](#) (class in [zyte_common_items](#)), 42
[regularPrice](#) ([zyte_common_items.Product](#) attribute), 19
[regularPrice](#) ([zyte_common_items.ProductFromList](#) attribute), 24
[regularPrice](#) ([zyte_common_items.ProductVariant](#) attribute), 22
[remoteStatus](#) ([zyte_common_items.JobPosting](#) attribute), 36
[rentalPeriod](#) ([zyte_common_items.RealEstate](#) attribute), 34
[reposts](#) ([zyte_common_items.Reactions](#) attribute), 51
[Request](#) (class in [zyte_common_items](#)), 51
[request\(\)](#) ([zyte_common_items.SearchRequestTemplate](#) method), 38
[RequestT](#) (class in [zyte_common_items.components.request](#)), 54
[requirements](#) ([zyte_common_items.JobPosting](#) attribute), 36
[reservationAction](#) ([zyte_common_items.BusinessPlace](#) attribute), 32
[reviewCount](#) ([zyte_common_items.AggregateRating](#) attribute), 47
[reviewSites](#) ([zyte_common_items.BusinessPlace](#) attribute), 32
- ## S
- [SearchRequestTemplate](#) (class in [zyte_common_items](#)), 38
[SearchRequestTemplateMetadata](#) (class in [zyte_common_items](#)), 39
[SearchRequestTemplatePage](#) (class in [zyte_common_items](#)), 43
[searchText](#) ([zyte_common_items.BusinessPlaceMetadata](#) attribute), 32
[searchText](#) ([zyte_common_items.Metadata](#) attribute), 53
[searchText](#) ([zyte_common_items.SocialMediaPostMetadata](#) attribute), 38
[simple_price_processor\(\)](#) (in module [zyte_common_items.processors](#)), 46
[size](#) ([zyte_common_items.Product](#) attribute), 19

size (zyte_common_items.ProductVariant attribute), 22
sku (zyte_common_items.Product attribute), 19
sku (zyte_common_items.ProductVariant attribute), 22
SocialMediaPost (class in zyte_common_items), 37
SocialMediaPostAuthor (class in zyte_common_items), 52
SocialMediaPostMetadata (class in zyte_common_items), 38
SocialMediaPostPage (class in zyte_common_items), 42
StarRating (class in zyte_common_items), 52
starRating (zyte_common_items.BusinessPlace attribute), 32
streetAddress (zyte_common_items.Address attribute), 47
style (zyte_common_items.Product attribute), 19
style (zyte_common_items.ProductVariant attribute), 22
subCategories (zyte_common_items.ArticleNavigation attribute), 30
subCategories (zyte_common_items.ProductNavigation attribute), 25

T

tags (zyte_common_items.BusinessPlace attribute), 32
telephone (zyte_common_items.BusinessPlace attribute), 32
text (zyte_common_items.Link attribute), 50
text (zyte_common_items.SocialMediaPost attribute), 37
timezone (zyte_common_items.BusinessPlace attribute), 32
to_probability_request_list() (in module zyte_common_items.converters), 54
to_probability_request_list_optional() (in module zyte_common_items.converters), 54
to_scrapy() (zyte_common_items.Request method), 51
tradeType (zyte_common_items.RealEstate attribute), 34
type (zyte_common_items.Gtin attribute), 49

U

unitCode (zyte_common_items.RealEstateArea attribute), 51
Url (class in zyte_common_items), 52
url (zyte_common_items.Article attribute), 27
url (zyte_common_items.ArticleFromList attribute), 29
url (zyte_common_items.ArticleList attribute), 28
url (zyte_common_items.ArticleNavigation attribute), 30
url (zyte_common_items.Audio attribute), 47
url (zyte_common_items.Author attribute), 48
url (zyte_common_items.Breadcrumb attribute), 49
url (zyte_common_items.BusinessPlace attribute), 32
url (zyte_common_items.Image attribute), 49
url (zyte_common_items.JobPosting attribute), 36

url (zyte_common_items.Link attribute), 50
url (zyte_common_items.NamedLink attribute), 50
url (zyte_common_items.Product attribute), 20
url (zyte_common_items.ProductFromList attribute), 24
url (zyte_common_items.ProductList attribute), 23
url (zyte_common_items.ProductNavigation attribute), 25
url (zyte_common_items.ProductVariant attribute), 22
url (zyte_common_items.RealEstate attribute), 34
url (zyte_common_items.Request attribute), 51
url (zyte_common_items.SearchRequestTemplate attribute), 39
url (zyte_common_items.SocialMediaPost attribute), 38
url (zyte_common_items.Video attribute), 52
url_to_str() (in module zyte_common_items.converters), 54
url_to_str_optional() (in module zyte_common_items.converters), 54

V

validationMessages (zyte_common_items.ArticleListMetadata attribute), 29
validationMessages (zyte_common_items.ArticleMetadata attribute), 27
validationMessages (zyte_common_items.ArticleNavigationMetadata attribute), 30
validationMessages (zyte_common_items.BusinessPlaceMetadata attribute), 32
validationMessages (zyte_common_items.components.metadata.Details attribute), 54
validationMessages (zyte_common_items.components.metadata.ListMetadata attribute), 53
validationMessages (zyte_common_items.JobPostingMetadata attribute), 37
validationMessages (zyte_common_items.Metadata attribute), 53
validationMessages (zyte_common_items.ProductListMetadata attribute), 24
validationMessages (zyte_common_items.ProductMetadata attribute), 22
validationMessages (zyte_common_items.ProductNavigationMetadata attribute), 25
validationMessages (zyte_common_items.RealEstateMetadata attribute), 35
validationMessages (zyte_common_items.SearchRequestTemplateMetadata attribute), 39
validationMessages (zyte_common_items.SocialMediaPostMetadata attribute), 38
validThrough (zyte_common_items.JobPosting attribute), 36
validThroughRaw (zyte_common_items.JobPosting attribute), 36
value (zyte_common_items.AdditionalProperty attribute), 46

`value` (*zyte_common_items.Amenity attribute*), 47
`value` (*zyte_common_items.Gtin attribute*), 49
`value` (*zyte_common_items.Header attribute*), 49
`value` (*zyte_common_items.RealEstateArea attribute*), 51
`valueMax` (*zyte_common_items.BaseSalary attribute*), 48
`valueMin` (*zyte_common_items.BaseSalary attribute*), 48
`variants` (*zyte_common_items.Product attribute*), 20
`Video` (class in *zyte_common_items*), 52
`videos` (*zyte_common_items.Article attribute*), 27
`virtualTourUrl` (*zyte_common_items.RealEstate attribute*), 34

W

`website` (*zyte_common_items.BusinessPlace attribute*), 32

Y

`yearBuilt` (*zyte_common_items.RealEstate attribute*), 34

Z

`zyte_common_items.converters`
module, 54

`ZyteItemAdapter` (class in *zyte_common_items*), 55

`ZyteItemKeepEmptyAdapter` (class in *zyte_common_items*), 55