
octo-api

Release 0.1.1

Python interface to the Octopus Energy API.

Dominic Davis-Foster

Apr 29, 2024

Contents

1	Installation	1
1.1	from PyPI	1
1.2	from GitHub	1
2	API Reference	3
2.1	octo_api.api	3
2.2	octo_api.consumption	8
2.3	octo_api.pagination	9
2.4	octo_api.products	11
2.5	octo_api.utils	19
3	Contributing	23
3.1	Overview	23
3.2	Coding style	23
3.3	Automated tests	23
3.4	Type Annotations	23
3.5	Build documentation locally	24
3.6	License	24
3.7	Downloading source code	25
	Python Module Index	27
	Index	29

Installation

1.1 from PyPI

```
$ python3 -m pip install octo-api --user
```

1.2 from GitHub

```
$ python3 -m pip install git+https://github.com/domdfcoding/octo-api@master --user
```


API Reference

2.1 octo_api.api

The primary interface to the Octopus Energy API.

Note: The Octopus Energy API uses the term “Grid Supply Point” (GSP) to refer to what are actually the 14 former Public Electricity Suppliers. The GSP terminology has been used here to better reflect the REST API.

Classes:

<code>OctoAPI(api_key)</code>	The primary interface to the Octopus Energy API.
-------------------------------	--

class `OctoAPI` (*api_key*)

Bases: `object`

The primary interface to the Octopus Energy API.

Parameters `api_key` (`str`) – API key to access the Octopus Energy API.

If you are an Octopus Energy customer, you can generate an API key from your [online dashboard](#).

Attributes:

<code>API_BASE</code>	The base URL of the Octopus Energy API.
<code>API_KEY</code>	The API key to access the Octopus Energy API.

Methods:

<code>get_consumption(mpan, serial_number, fuel[, ...])</code>	Return a list of consumption values for half-hour periods for a given meter-point and meter.
<code>get_grid_supply_point(postcode)</code>	Returns the grid supply point for the given postcode.
<code>get_meter_point_details(mpan)</code>	Retrieve the details of a meter-point.
<code>get_product_info(product_code[, ...])</code>	Retrieve the details of a product (including all its tariffs) for a particular point in time.
<code>get_products([is_variable, is_green, ...])</code>	Returns a list of energy products.
<code>get_tariff_charges(product_code, ...[, ...])</code>	Returns a list of time periods and their associated unit rates charges.

`API_BASE`

Type: `SlumberURL`

The base URL of the Octopus Energy API.

API_KEY**Type:** `Secret`

The API key to access the Octopus Energy API.

get_consumption (*mpan*, *serial_number*, *fuel*, *period_from*=None, *period_to*=None, *page_size*=100, *reverse*=False, *group_by*=None)

Return a list of consumption values for half-hour periods for a given meter-point and meter.

Unit of measurement:

- Electricity meters: kWh
- SMETS1 Secure gas meters: kWh
- SMETS2 gas meters: m³

Attention: Half-hourly consumption data is only available for smart meters. Requests for consumption data for non-smart meters will return an empty response payload.

Parameters

- **mpan** (`str`) – The electricity meter-point’s MPAN or gas meter-point’s MPRN.
- **serial_number** (`str`) – The meter’s serial number.
- **fuel** (`Literal['electricity', 'gas']`)
- **period_from** (`Optional[datetime]`) – Show consumption for periods which start at or after the given datetime. This parameter can be provided on its own. Default `None`.
- **period_to** (`Optional[datetime]`) – Show consumption for periods which start at or before the given datetime. This parameter also requires providing the `period_from` parameter to create a range. Default `None`.
- **page_size** (`int`) – Page size of returned results. Default is 100, maximum is 25,000 to give a full year of half-hourly consumption details.
- **reverse** (`bool`) – Returns the results ordered from most oldest to newest. By default the results are from most recent backwards.
- **group_by** (`Optional[str]`) – The grouping of the consumption data. By default the consumption is returned in half-hour periods.

Possible alternatives are:

- `'hour'`
- `'day'`
- `'week'`
- `'month'`
- `'quarter'`

Return type `PaginatedResponse[Consumption]`

get_grid_supply_point (*postcode*)

Returns the grid supply point for the given postcode.

Parameters `postcode` (`str`)

Raises `ValueError` if the postcode cannot be mapped to a GSP.

Return type `Region`

get_meter_point_details (`mpan`)

Retrieve the details of a meter-point.

This can be used to get the GSP of a given meter-point.

Parameters `mpan` (`str`) – The electricity meter-point's MPAN.

Return type `MeterPointDetails`

Returns

get_product_info (`product_code`, `tariffs_active_at=None`)

Retrieve the details of a product (including all its tariffs) for a particular point in time.

Parameters

- **product_code** (`str`) – The code of the product to be retrieved, for example VAR-17-01-11.
- **tariffs_active_at** (`Optional[datetime]`) – The point in time in which to show the active charges. Defaults to current datetime. Default `None`.

Example

```
>>> api.get_product_info(product_code='VAR-17-01-11')
octo_api.products.DetailedProduct(
    available_from='2017-01-11T10:00:00+00:00',
    available_to='2018-02-15T00:00:00+00:00',
    brand='S_ENERGY',
    code='7-01-11',
    description='This variable tariff always offers great value - driven by
↳our'
                                'belief that prices should be fair for the long term, not
↳just a'
                                'fixed term. We aim for 50% renewable electricity on this
↳tariff.',
    display_name='pus',
    full_name='ctopus January 2017 v1',
    is_business=False,
    is_green=False,
    is_prepay=False,
    is_restricted=False,
    is_tracker=False,
    is_variable=True,
    links=[
        {
            'href': 'https://api.octopus.energy/v1/products/VAR-17-01-11/',
            'method': 'GET',
            'rel': 'self'
        }
    ],
    term=None,
    tariffs_active_at='2020-10-26T11:15:17.208285+00:00',
    single_register_electricity_tariffs=RegionalTariffs(['direct_debit_monthly
↳']),
```

(continues on next page)

(continued from previous page)

```

    dual_register_electricity_tariffs=RegionalTariffs(['direct_debit_monthly
↪']),
    single_register_gas_tariffs=RegionalTariffs(['direct_debit_monthly']),
    sample_quotes=RegionalQuotes([dual_fuel_dual_rate, dual_fuel_single_rate, ↪
↪electricity_dual_rate, electricity_single_rate]),
    sample_consumption={
        'electricity_single_rate': {'electricity_standard': 2900},
        'electricity_dual_rate': {
            'electricity_day': 2436,
            'electricity_night': 1764
        },
        'dual_fuel_single_rate': {
            'electricity_standard': 2900,
            'gas_standard': 12000
        },
        'dual_fuel_dual_rate': {
            'electricity_day': 2436,
            'electricity_night': 1764,
            'gas_standard': 12000
        }
    },
),

```

Return type *DetailedProduct*

get_products (*is_variable=None, is_green=None, is_tracker=None, is_prepay=None, is_business=False, available_at=None*)

Returns a list of energy products.

By default, the results only include public energy products. Authenticated organisations will also see products available to their organisation.

Parameters

- **is_variable** (Optional[bool]) – Show only variable products. Default *None*.
- **is_green** (Optional[bool]) – Show only green products. Default *None*.
- **is_tracker** (Optional[bool]) – Show only tracker products. Default *None*.
- **is_prepay** (Optional[bool]) – Show only pre-pay products. Default *None*.
- **is_business** (bool) – Show only business products. Default *False*.
- **available_at** (Optional[datetime]) – Show products available for new agreements on the given datetime. Defaults to the current datetime, effectively showing products that are currently available.

Example

```

>>> api.get_products() [0]
octo_api.products.Product (
    available_from='2016-01-01T:00:00:00+00:00',
    available_to=None,
    brand='AFFECT_ENERGY',
    code='1201',
    description='Affect Standard Tariff',

```

(continues on next page)

(continued from previous page)

```

display_name='Affect Standard Tariff',
full_name='Affect Standard Tariff',
is_business=False,
is_green=False,
is_prepay=False,
is_restricted=False,
is_tracker=False,
is_variable=True,
links=[
    {
        'href': 'https://api.octopus.energy/v1/products/1201/',
        'method': 'GET',
        'rel': 'self'
    }
],
term=None,
direction='IMPORT',
)

```

Return type `PaginatedResponse[Product]`

get_tariff_charges (*product_code*, *tariff_code*, *fuel*, *rate_type*, *period_from*=None, *period_to*=None, *page_size*=100)

Returns a list of time periods and their associated unit rates charges.

If the tariff has a fixed unit rate the list will only contain one element.

Parameters

- **product_code** (`str`) – The code of the product to be retrieved, for example VAR-17-01-11.
- **tariff_code** (`str`) – The code of the tariff to be retrieved, for example E-1R-VAR-17-01-11-A. From what I can tell the format is:

```
<E for electricity><optional hyphen><1R for single rate?><the product_
↪code>-<the grid supply point>
```

- **fuel** (`Literal['electricity', 'gas']`)
- **rate_type** (`RateType`)
- **period_from** (`Optional[datetime]`) – Show charges active from the given datetime (inclusive). This parameter can be provided on its own. Default `None`.
- **period_to** (`Optional[datetime]`) – Show charges active up to the given datetime (exclusive). You must also provide the `period_from` parameter in order to create a range. Default `None`.
- **page_size** (`int`) – Page size of returned results. Default is 100, maximum is 1,500 to give up to a month of half-hourly prices.

Note: If you're using this API to query future unit-rates of the Agile Octopus product, note that day-ahead prices are normally created by 4pm in the Europe/London timezone. Further, the market index used to calculate unit rates is based in the CET timezone (UTC+1) and so its "day" corresponds to 11pm to

11pm in UK time. Hence, if you query today's unit rates before 4pm, you'll get 46 results back rather than 48.

Return type *PaginatedResponse*[*RateInfo*]

2.2 octo_api.consumption

Class to represent consumption data.

Classes:

<i>Consumption</i> (consumption, interval_start, ...)	Represents the consumption for a given period of time.
---	--

class *Consumption* (consumption, interval_start, interval_end)

Bases: *object*

Represents the consumption for a given period of time.

Parameters

- **consumption** (*float*) – The consumption.
- **interval_start** (*Union*[*str*, *datetime*, *None*]) – The start of the time period.
- **interval_end** (*Union*[*str*, *datetime*, *None*]) – The end of the time period.

Attributes:

<i>consumption</i>	The consumption.
<i>interval_end</i>	The end of the time period.
<i>interval_start</i>	The start of the time period.

Methods:

<i>from_dict</i> (d)	Construct an instance of <i>Consumption</i> from a dictionary.
<i>to_dict</i> ([convert_values])	Returns a dictionary containing the contents of the <i>Consumption</i> object.

consumption

Type: *float*

The consumption.

classmethod *from_dict* (d)

Construct an instance of *Consumption* from a dictionary.

Parameters *d* (*Mapping*[*str*, *Any*]) – The dictionary.

interval_end

Type: *datetime*

The end of the time period.

interval_start

Type: `datetime`

The start of the time period.

to_dict (*convert_values=False*)

Returns a dictionary containing the contents of the *Consumption* object.

Parameters **convert_values** (`bool`) – Recursively convert values into dictionaries, lists etc. as appropriate. Default `False`.

Return type `MutableMapping[str, Any]`

2.3 octo_api.pagination

Class for handling paginated API responses.

Classes:

<i>OctoResponse</i>	<code>TypedDict</code> representing the raw JSON data returned by the Octopus Energy API.
<i>PaginatedResponse</i> (<i>query_url</i> [, <i>query_params</i> , ...])	Represents a multi-page response from a REST API.

typeddict OctoResponse

Bases: `TypedDict`

`TypedDict` representing the raw JSON data returned by the Octopus Energy API.

Required Keys

- **count** (`int`) – The total number of responses.
- **next** (`str`) – The URL of the next page of results.
- **previous** (`str`) – The URL of the previous page of results.
- **results** (`List[Dict[str, Any]]`) – The current page of results.

class PaginatedResponse (*query_url*, *query_params=None*, *obj_type=<class 'dict'>*)

Bases: `Iterable[~T]`

Represents a multi-page response from a REST API.

The items within the response can be iterated over or accessed by their indices. The total number of items can be accessed with `len(response)`.

Parameters

- **query_url** (`SlumberURL`) – The initial query URL.
- **query_params** (`Optional[MutableMapping[str, Any]]`) – The parameters to the query. Default `None`.
- **obj_type** (`Type`) – The object to convert the response data to. Default `dict`.

Note: This class assumes the JSON response is in the format used by [Django REST framework](#).

The response should be in the following format:

```
{
  "count": 1023,
  "next": "https://api.example.org/accounts/?page=5",
  "previous": "https://api.example.org/accounts/?page=3",
  "results": []
}
```

See <https://www.django-rest-framework.org/api-guide/pagination/> for more information.

Methods:

<code>__eq__(other)</code>	Return <code>self == other</code> .
<code>__getitem__(item)</code>	Returns the item or items in the <i>PaginatedResponse</i> , as given by the index or slice.
<code>__iter__()</code>	Iterate over items in the <i>PaginatedResponse</i> .
<code>__len__()</code>	Returns the number of items in the <i>PaginatedResponse</i> .

`__eq__(other)`
Return `self == other`.

Return type `bool`

`__getitem__(item)`
Returns the item or items in the *PaginatedResponse*, as given by the index or slice.

Parameters `item` (`Union[int, slice]`)

Return type `Union[~_T, List[~_T]]`

Overloads

- `__getitem__(item: int) -> ~_T`
- `__getitem__(item: slice) -> List[~_T]`

`__iter__()`
Iterate over items in the *PaginatedResponse*.

Return type `Iterator[~_T]`

`__len__()`
Returns the number of items in the *PaginatedResponse*.

Return type `int`

2.4 octo_api.products

Classes to model products and tariffs.

Classes:

<i>BaseProduct</i> (available_from, available_to, ...)	Represents an Octopus Energy product.
<i>Product</i> (available_from, available_to, brand, ...)	Represents an Octopus Energy product.
<i>DetailedProduct</i> (available_from, ...)	Represents an Octopus Energy product, with detailed tariff information.
<i>Tariff</i> (code, standing_charge_exc_vat, ...[, ...])	Represents a tariff for a product.
<i>RateInfo</i> (value_exc_vat, value_inc_vat, ...)	Represents the unit rate of a tariff at a particular period in time.
<i>RegionalTariffs</i>	Mapping of GSP regions to a mapping of payment methods to <i>Tariffs</i> .
<i>RegionalQuotes</i>	Mapping of GSP regions to a mapping of payment methods to a mapping of fuel types to <i>Quotes</i> .

class BaseProduct (*available_from, available_to, brand, code, description, display_name, full_name, is_business, is_green, is_prepay, is_restricted, is_tracker, is_variable, links, term*)

Bases: `object`

Represents an Octopus Energy product.

Parameters

- **available_from** (`Union[str, datetime, None]`) – The date from which the product is available.
- **available_to** (`Union[str, datetime, None]`) – The date until which the product is available.
- **brand** (`str`) – The brand under which the product is sold.
- **code** (`str`) – The tariff code.
- **description** (`str`) – A description of the product.
- **display_name** (`str`) – The display name of the product.
- **full_name** (`str`) – The name of the product.
- **is_business** (`bool`) – Whether the product is for businesses.
- **is_green** (`bool`) – Whether the product is green.
- **is_prepay** (`bool`) – Whether the product is prepay.
- **is_restricted** (`bool`) – Whether the product is restricted.
- **is_tracker** (`bool`) – Whether the product tracks the wholesale electricity rate.
- **is_variable** (`bool`) – Whether the product has a variable tariff.
- **links** (`Iterable[MutableMapping[str, Any]]`) – Links associated with this product.
- **term** (`Optional[int]`) – The number of months that a product lasts for if it is fixed length.

Attributes:

<i>available_from</i>	The date from which the product is available.
<i>available_to</i>	The date until which the product is available.
<i>brand</i>	The brand under which the product is sold.
<i>code</i>	The code of the product.
<i>description</i>	A description of the product.
<i>display_name</i>	The display name of the product.
<i>full_name</i>	The name of the product.
<i>is_business</i>	Whether the product is for businesses.
<i>is_green</i>	Whether the product is green.
<i>is_prepay</i>	Whether the product is prepay.
<i>is_restricted</i>	Whether the product is restricted.
<i>is_tracker</i>	Whether the product tracks the wholesale electricity rate.
<i>is_variable</i>	Whether the product has a variable tariff.
<i>links</i>	Links associated with this product.
<i>term</i>	The number of months that a product lasts for if it is fixed length.

Methods:

<i>from_dict</i> (d)	Construct an instance of <i>BaseProduct</i> from a dictionary.
<i>to_dict</i> ([convert_values])	Returns a dictionary containing the contents of the <i>BaseProduct</i> object.

available_from**Type:** `Optional[datetime]`

The date from which the product is available.

available_to**Type:** `Optional[datetime]`

The date until which the product is available.

brand**Type:** `str`

The brand under which the product is sold.

code**Type:** `str`

The code of the product.

description**Type:** `str`

A description of the product.

display_name**Type:** `str`

The display name of the product.

classmethod `from_dict(d)`

Construct an instance of *BaseProduct* from a dictionary.

Parameters `d` (`Mapping[str, Any]`) – The dictionary.

full_name

Type: `str`

The name of the product.

is_business

Type: `bool`

Whether the product is for businesses.

is_green

Type: `bool`

Whether the product is green.

is_prepay

Type: `bool`

Whether the product is prepay.

is_restricted

Type: `bool`

Whether the product is restricted.

is_tracker

Type: `bool`

Whether the product tracks the wholesale electricity rate.

is_variable

Type: `bool`

Whether the product has a variable tariff.

links

Type: `List[MutableMapping[str, Any]]`

Links associated with this product.

term

Type: `Optional[int]`

The number of months that a product lasts for if it is fixed length.

to_dict (`convert_values=False`)

Returns a dictionary containing the contents of the *BaseProduct* object.

Parameters `convert_values` (`bool`) – Recursively convert values into dictionaries, lists etc. as appropriate. Default `False`.

Return type `MutableMapping[str, Any]`

```
class Product (available_from, available_to, brand, code, description, display_name, full_name,  
               is_business, is_green, is_prepay, is_restricted, is_tracker, is_variable, links, term,  
               direction)
```

Bases: `octo_api.products.BaseProduct`

Represents an Octopus Energy product.

Parameters

- **available_from** (`Union[str, datetime, None]`) – The date from which the product is available.
- **available_to** (`Union[str, datetime, None]`) – The date until which the product is available.
- **brand** (`str`) – The brand under which the product is sold.
- **code** (`str`) – The tariff code.
- **description** (`str`) – A description of the product.
- **display_name** (`str`) – The display name of the product.
- **full_name** (`str`) – The name of the product.
- **is_business** (`bool`) – Whether the product is for businesses.
- **is_green** (`bool`) – Whether the product is green.
- **is_prepay** (`bool`) – Whether the product is prepay.
- **is_restricted** (`bool`) – Whether the product is restricted.
- **is_tracker** (`bool`) – Whether the product tracks the wholesale electricity rate.
- **is_variable** (`bool`) – Whether the product has a variable tariff.
- **links** (`Iterable[MutableMapping[str, Any]]`) – Links associated with this product.
- **term** (`Optional[int]`) – The number of months that a product lasts for if it is fixed length.
- **direction** (`str`) – The direction of the product (supply to the customer or supply to the grid).

Attributes:

<code>direction</code>	The direction of the product (supply to the customer or supply to the grid).
------------------------	--

`direction`

Type: `str`

The direction of the product (supply to the customer or supply to the grid).

```
class DetailedProduct (available_from, available_to, brand, code, description, display_name,  
                        full_name, is_business, is_green, is_prepay, is_restricted, is_tracker,  
                        is_variable, links, term, tariffs_active_at, single_register_electricity_tariffs,  
                        dual_register_electricity_tariffs, single_register_gas_tariffs, sample_quotes,  
                        sample_consumption)
```

Bases: `octo_api.products.BaseProduct`

Represents an Octopus Energy product, with detailed tariff information.

Each `*_tariffs` object will have up to 14 keys; one for each GSP. For each GSP the applicable tariffs are

listed under their associated payment method, e.g. `direct_debit_monthly`.

- The `standard_unit_rate_*` values are listed in p/kWh (pence per kilowatt hour).
- The `standing_charge_*` values are listed in p/day (pence per day).
- The `annual_cost_*` values are listed in p (pence).

Parameters

- **available_from** (`Union[str, datetime, None]`) – The date from which the product is available.
- **available_to** (`Union[str, datetime, None]`) – The date until which the product is available.
- **brand** (`str`) – The brand under which the product is sold.
- **code** (`str`) – The tariff code.
- **description** (`str`) – A description of the product.
- **display_name** (`str`) – The display name of the product.
- **full_name** (`str`) – The name of the product.
- **is_business** (`bool`) – Whether the product is for businesses.
- **is_green** (`bool`) – Whether the product is green.
- **is_prepay** (`bool`) – Whether the product is prepay.
- **is_restricted** (`bool`) – Whether the product is restricted.
- **is_tracker** (`bool`) – Whether the product tracks the wholesale electricity rate.
- **is_variable** (`bool`) – Whether the product has a variable tariff.
- **links** (`Iterable[MutableMapping[str, Any]]`) – Links associated with this product.
- **term** (`Optional[int]`) – The number of months that a product lasts for if it is fixed length.
- **tariffs_active_at** (`Union[str, datetime, None]`)
- **single_register_electricity_tariffs** (`Dict[str, Dict[str, Dict[str, Any]]]`) – Mapping of GSPs to applicable tariffs for each payment method, e.g. `direct_debit_monthly`.
- **dual_register_electricity_tariffs** (`Dict[str, Dict[str, Dict[str, Any]]]`) – Mapping of GSPs to applicable tariffs for each payment method, e.g. `direct_debit_monthly`.
- **single_register_gas_tariffs** (`Dict[str, Dict[str, Dict[str, Any]]]`) – Mapping of GSPs to applicable tariffs for each payment method, e.g. `direct_debit_monthly`.
- **sample_quotes** (`Dict[str, Dict[str, Dict[str, Any]]]`)
- **sample_consumption** (`Dict`)

Attributes:

<code>dual_register_electricity_tariffs</code>	Mapping of GSPs to applicable tariffs for each payment method, e.g.
<code>sample_consumption</code>	
<code>sample_quotes</code>	

continues on next page

Table 13 – continued from previous page

<code>single_register_electricity_tariffs</code>	Mapping of GSPs to applicable tariffs for each payment method, e.g.
<code>single_register_gas_tariffs</code>	Mapping of GSPs to applicable tariffs for each payment method, e.g.
<code>tariffs_active_at</code>	

Methods:

<code>from_dict(d)</code>	Construct an instance of <i>DetailedProduct</i> from a dictionary.
<code>to_dict([convert_values])</code>	Returns a dictionary containing the contents of the <i>DetailedProduct</i> object.

dual_register_electricity_tariffs**Type:** Dict

Mapping of GSPs to applicable tariffs for each payment method, e.g. direct_debit_monthly.

classmethod from_dict (d)Construct an instance of *DetailedProduct* from a dictionary.**Parameters** **d** (Mapping[str, Any]) – The dictionary.**sample_consumption****Type:** Dict**sample_quotes****Type:** Dict**single_register_electricity_tariffs****Type:** Dict

Mapping of GSPs to applicable tariffs for each payment method, e.g. direct_debit_monthly.

single_register_gas_tariffs**Type:** Dict

Mapping of GSPs to applicable tariffs for each payment method, e.g. direct_debit_monthly.

tariffs_active_at**Type:** Optional[datetime]**to_dict (convert_values=False)**Returns a dictionary containing the contents of the *DetailedProduct* object.**Parameters** **convert_values** (bool) – Recursively convert values into dictionaries, lists etc. as appropriate. Default False.**Return type** MutableMapping[str, Any]**class RateInfo (value_exc_vat, value_inc_vat, valid_from, valid_to)**

Bases: object

Represents the unit rate of a tariff at a particular period in time.

Parameters

- **value_exc_vat** (`float`) – In p/kWh (pence per kilowatt hour).
- **value_inc_vat** (`float`) – In p/kWh (pence per kilowatt hour).
- **valid_from** (`Union[str, datetime, None]`) – The date and time from which this rate is in effect.
- **valid_to** (`Union[str, datetime, None]`) – The date and time until which this rate is in effect, or `None` if this rate continues in perpetuity.

Methods:

<code>from_dict(d)</code>	Construct an instance of <i>RateInfo</i> from a dictionary.
<code>to_dict([convert_values])</code>	Returns a dictionary containing the contents of the <i>RateInfo</i> object.

Attributes:

<code>valid_from</code>	The date and time from which this rate is in effect.
<code>valid_to</code>	The date and time until which this rate is in effect, or <code>None</code> if this rate continues in perpetuity.
<code>value_exc_vat</code>	In p/kWh (pence per kilowatt hour).
<code>value_inc_vat</code>	In p/kWh (pence per kilowatt hour).

classmethod from_dict (d)

Construct an instance of *RateInfo* from a dictionary.

Parameters **d** (`Mapping[str, Any]`) – The dictionary.

to_dict (convert_values=False)

Returns a dictionary containing the contents of the *RateInfo* object.

Parameters **convert_values** (`bool`) – Recursively convert values into dictionaries, lists etc. as appropriate. Default `False`.

Return type `MutableMapping[str, Any]`

valid_from

Type: `datetime`

The date and time from which this rate is in effect.

valid_to

Type: `Optional[datetime]`

The date and time until which this rate is in effect, or `None` if this rate continues in perpetuity.

value_exc_vat

Type: `float`

In p/kWh (pence per kilowatt hour).

value_inc_vat

Type: `float`

In p/kWh (pence per kilowatt hour).

```
class Tariff (code, standing_charge_exc_vat, standing_charge_inc_vat, online_discount_exc_vat,  
              online_discount_inc_vat, dual_fuel_discount_exc_vat, dual_fuel_discount_inc_vat,  
              exit_fees_exc_vat, exit_fees_inc_vat, links, standard_unit_rate_exc_vat=None,  
              standard_unit_rate_inc_vat=None, day_unit_rate_exc_vat=None,  
              day_unit_rate_inc_vat=None, night_unit_rate_exc_vat=None,  
              night_unit_rate_inc_vat=None)
```

Bases: `object`

Represents a tariff for a product.

Parameters

- **code** (`str`) – The tariff code.
- **standing_charge_exc_vat** (`float`) – In p/day (pence per day).
- **standing_charge_inc_vat** (`float`) – In p/day (pence per day).
- **online_discount_exc_vat** (`int`)
- **online_discount_inc_vat** (`int`)
- **dual_fuel_discount_exc_vat** (`int`)
- **dual_fuel_discount_inc_vat** (`int`)
- **exit_fees_exc_vat** (`int`)
- **exit_fees_inc_vat** (`int`)
- **links** (`List[Dict[str, Any]]`) – Links associated with this product.
- **standard_unit_rate_exc_vat** (`Optional[float]`) – In p/kWh (pence per kilowatt hour). Default `None`.
- **standard_unit_rate_inc_vat** (`Optional[float]`) – In p/kWh (pence per kilowatt hour). Default `None`.
- **day_unit_rate_exc_vat** (`Optional[float]`) – In p/kWh (pence per kilowatt hour). Default `None`.
- **day_unit_rate_inc_vat** (`Optional[float]`) – In p/kWh (pence per kilowatt hour). Default `None`.
- **night_unit_rate_exc_vat** (`Optional[float]`) – In p/kWh (pence per kilowatt hour). Default `None`.
- **night_unit_rate_inc_vat** (`Optional[float]`) – In p/kWh (pence per kilowatt hour). Default `None`.

Methods:

<code>from_dict(d)</code>	Construct an instance of <code>Tariff</code> from a dictionary.
<code>to_dict([convert_values])</code>	Returns a dictionary containing the contents of the <code>Tariff</code> object.

classmethod `from_dict(d)`

Construct an instance of *Tariff* from a dictionary.

Parameters *d* (*Mapping*[*str*, *Any*]) – The dictionary.

to_dict (*convert_values=False*)

Returns a dictionary containing the contents of the *Tariff* object.

Parameters *convert_values* (*bool*) – Recursively convert values into dictionaries, lists etc. as appropriate. Default *False*.

Return type *MutableMapping*[*str*, *Any*]

class *RegionalTariffs*

Bases: *Dict*[*str*, *Dict*[*str*, *Tariff*]]

Mapping of GSP regions to a mapping of payment methods to *Tariffs*.

Methods:

<code>__str__()</code>	Return <i>str</i> (<i>self</i>).
------------------------	------------------------------------

`__str__()`

Return *str*(*self*).

Return type *str*

2.5 octo_api.utils

Utility functions.

Classes:

<i>MeterPointDetails</i> (<i>mpan</i> , <i>gsp</i> , <i>profile_class</i>)	Information about a meter point.
<i>RateType</i> (<i>value</i>)	Enumeration of different rate types.
<i>Region</i> (<i>value</i>)	Enumeration of different electricity supply regions.

Functions:

<i>add_repr</i> (<i>cls</i>)	Add a pretty-printed <code>__repr__</code> function to the decorated <i>attrs</i> class.
<i>from_iso_zulu</i> (<i>the_datetime</i>)	Constructs a <code>datetime.datetime</code> object from an ISO 8601 format string.

namedtuple *MeterPointDetails* (*mpan*, *gsp*, *profile_class*)

Bases: *NamedTuple*

Information about a meter point.

Fields

0) *mpan* (*str*) – The meter point access number.

1) *gsp* (*Region*) – The grid supply point/region that the meter point is located in.

2) **profile_class** (*int*) – The profile class of the meter point.

- **Profile Class 1** – Domestic Unrestricted Customers
- **Profile Class 2** – Domestic Economy 7 Customers
- **Profile Class 3** – Non-Domestic Unrestricted Customers
- **Profile Class 4** – Non-Domestic Economy 7 Customers
- **Profile Class 5** – Non-Domestic Maximum Demand (MD) Customers with a Peak Load Factor (LF) of less than 20%
- **Profile Class 6** – Non-Domestic Maximum Demand Customers with a Peak Load Factor between 20% and 30%
- **Profile Class 7** – Non-Domestic Maximum Demand Customers with a Peak Load Factor between 30% and 40%
- **Profile Class 8** – Non-Domestic Maximum Demand Customers with a Peak Load Factor over 40%

Information from <https://www.elexon.co.uk/knowledgebase/profile-classes/>

See also:

[Load Profiles and their use in Electricity Settlement](#) by Elexon

__repr__ ()

Return a string representation of the *MeterPointDetails*.

Return type *str*

enum RateType (*value*)

Bases: *enum_tools.custom_enums.StrEnum*

Enumeration of different rate types.

Member Type *str*

Valid values are as follows:

StandingCharge = *<RateType.StandingCharge: 'standing-charges'>*

StandardUnitRate = *<RateType.StandardUnitRate: 'standard-unit-rates'>*

DayUnitRate = *<RateType.DayUnitRate: 'day-unit-rates'>*

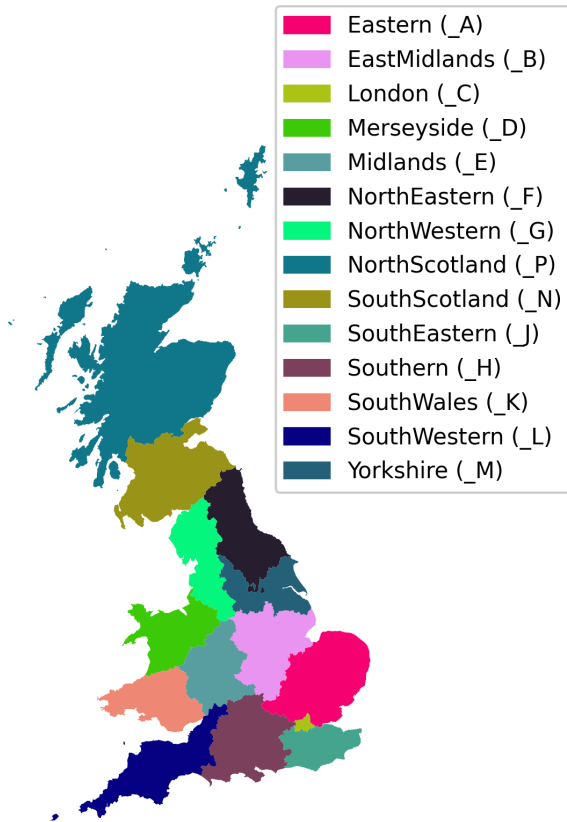
NightUnitRate = *<RateType.NightUnitRate: 'night-unit-rates'>*

enum Region (*value*)

Bases: *enum_tools.custom_enums.StrEnum*

Enumeration of different electricity supply regions.

The different regions can be seen on the following map:



Member Type `str`

Valid values are as follows:

Eastern = `<Region.Eastern: '_A'>`

EastMidlands = `<Region.EastMidlands: '_B'>`

London = `<Region.London: '_C'>`

Merseyside = `<Region.Merseyside: '_D'>`

Midlands = `<Region.Midlands: '_E'>`

NorthEastern = `<Region.NorthEastern: '_F'>`

NorthWestern = `<Region.NorthWestern: '_G'>`

Southern = `<Region.Southern: '_H'>`

SouthEastern = `<Region.SouthEastern: '_J'>`

SouthWales = `<Region.SouthWales: '_K'>`

SouthWestern = `<Region.SouthWestern: '_L'>`

```
Yorkshire = <Region.Yorkshire:  '_M'>
```

```
SouthScotland = <Region.SouthScotland:  '_N'>
```

```
NorthScotland = <Region.NorthScotland:  '_P'>
```

add_repr (*cls*)

Add a pretty-printed `__repr__` function to the decorated attrs class.

Parameters *cls* (*Type*)

See also:

`attr_utils.pprinter.pretty_repr()`.

Return type *Type*

from_iso_zulu (*the_datetime*)

Constructs a `datetime.datetime` object from an [ISO 8601](#) format string.

This function understands the character Z as meaning Zulu time (GMT/UTC).

Parameters *the_datetime* (`Union[str, datetime, None]`)

Return type `Optional[datetime]`

Contributing

3.1 Overview

`octo-api` uses `tox` to automate testing and packaging, and `pre-commit` to maintain code quality.

Install `pre-commit` with `pip` and install the git hook:

```
$ python -m pip install pre-commit
$ pre-commit install
```

3.2 Coding style

`formate` is used for code formatting.

It can be run manually via `pre-commit`:

```
$ pre-commit run formate -a
```

Or, to run the complete autoformatting suite:

```
$ pre-commit run -a
```

3.3 Automated tests

Tests are run with `tox` and `pytest`. To run tests for a specific Python version, such as Python 3.6:

```
$ tox -e py36
```

To run tests for all Python versions, simply run:

```
$ tox
```

3.4 Type Annotations

Type annotations are checked using `mypy`. Run `mypy` using `tox`:

```
$ tox -e mypy
```

3.5 Build documentation locally

The documentation is powered by Sphinx. A local copy of the documentation can be built with `tox`:

```
$ tox -e docs
```

3.6 License

octo-api is licensed under the [MIT License](#)

A short and simple permissive license with conditions only requiring preservation of copyright and license notices. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

Permissions

- Commercial use – The licensed material and derivatives may be used for commercial purposes.
- Modification – The licensed material may be modified.
- Distribution – The licensed material may be distributed.
- Private use – The licensed material may be used and modified in private.

Conditions

- License and copyright notice – A copy of the license and copyright notice must be included with the licensed material.

Limitations

- Liability – This license includes a limitation of liability.
- Warranty – This license explicitly states that it does NOT provide any warranty.

[See more information on choosealicense.com](#) ⇒

```
Copyright (c) 2020 Dominic Davis-Foster
```

```
Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
```

```
The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.
```

```
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR
OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE
OR OTHER DEALINGS IN THE SOFTWARE.
```

3.7 Downloading source code

The octo-api source code is available on GitHub, and can be accessed from the following URL: <https://github.com/domdfcoding/octo-api>

If you have git installed, you can clone the repository with the following command:

```
$ git clone https://github.com/domdfcoding/octo-api
```

```
Cloning into 'octo-api'...
remote: Enumerating objects: 47, done.
remote: Counting objects: 100% (47/47), done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 173 (delta 16), reused 17 (delta 6), pack-reused 126
Receiving objects: 100% (173/173), 126.56 KiB | 678.00 KiB/s, done.
Resolving deltas: 100% (66/66), done.
```

Alternatively, the code can be downloaded in a ‘zip’ file by clicking:

Clone or download → Download Zip

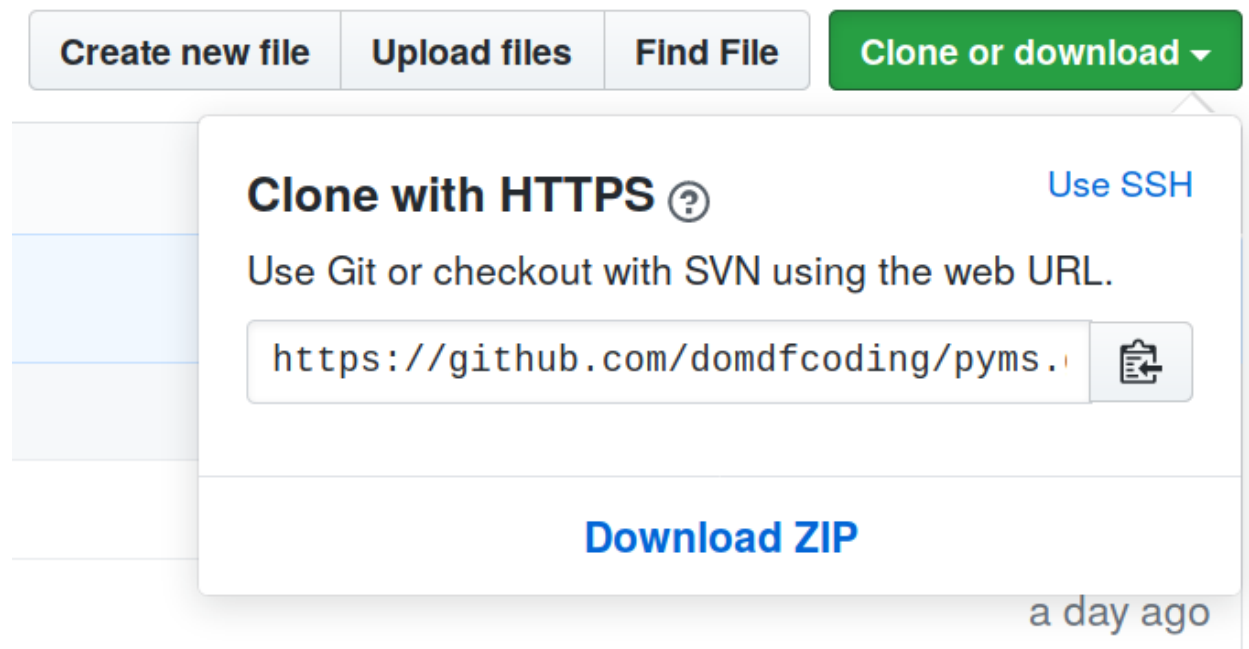


Fig. 1: Downloading a ‘zip’ file of the source code

3.7.1 Building from source

The recommended way to build `octo-api` is to use `tox`:

```
$ tox -e build
```

The source and wheel distributions will be in the directory `dist`.

If you wish, you may also use `pep517.build` or another **PEP 517**-compatible build tool.

Parts of this documentation based on <https://developer.octopus.energy/docs/api/>

© 2020, Kraken Technologies Ltd.

Python Module Index

O

- `octo_api.api`, 3
- `octo_api.consumption`, 8
- `octo_api.pagination`, 9
- `octo_api.products`, 11
- `octo_api.utils`, 19

Symbols

`__eq__()` (*PaginatedResponse* method), 10
`__getitem__()` (*PaginatedResponse* method), 10
`__iter__()` (*PaginatedResponse* method), 10
`__len__()` (*PaginatedResponse* method), 10
`__repr__()` (*MeterPointDetails* method), 20
`__str__()` (*RegionalTariffs* method), 19

A

`add_repr()` (in module *octo_api.utils*), 22
`API_BASE` (*OctoAPI* attribute), 3
`API_KEY` (*OctoAPI* attribute), 4
`available_from` (*BaseProduct* attribute), 12
`available_to` (*BaseProduct* attribute), 12

B

`BaseProduct` (class in *octo_api.products*), 11
`brand` (*BaseProduct* attribute), 12

C

`code` (*BaseProduct* attribute), 12
`Consumption` (class in *octo_api.consumption*), 8
`consumption` (*Consumption* attribute), 8

D

`DayUnitRate` (*RateType* attribute), 20
`description` (*BaseProduct* attribute), 12
`DetailedProduct` (class in *octo_api.products*), 14
`direction` (*Product* attribute), 14
`display_name` (*BaseProduct* attribute), 12
`dual_register_electricity_tariffs`
 (*DetailedProduct* attribute), 16

E

`Eastern` (*Region* attribute), 21
`EastMidlands` (*Region* attribute), 21

F

`from_dict()` (*BaseProduct* class method), 13
`from_dict()` (*Consumption* class method), 8
`from_dict()` (*DetailedProduct* class method), 16
`from_dict()` (*RateInfo* class method), 17
`from_dict()` (*Tariff* class method), 18

`from_iso_zulu()` (in module *octo_api.utils*), 22
`full_name` (*BaseProduct* attribute), 13

G

`get_consumption()` (*OctoAPI* method), 4
`get_grid_supply_point()` (*OctoAPI* method), 4
`get_meter_point_details()` (*OctoAPI* method), 5
`get_product_info()` (*OctoAPI* method), 5
`get_products()` (*OctoAPI* method), 6
`get_tariff_charges()` (*OctoAPI* method), 7
`gsp` (namedtuple field)
 MeterPointDetails (namedtuple in *octo_api.utils*), 19

I

`interval_end` (*Consumption* attribute), 8
`interval_start` (*Consumption* attribute), 9
`is_business` (*BaseProduct* attribute), 13
`is_green` (*BaseProduct* attribute), 13
`is_prepay` (*BaseProduct* attribute), 13
`is_restricted` (*BaseProduct* attribute), 13
`is_tracker` (*BaseProduct* attribute), 13
`is_variable` (*BaseProduct* attribute), 13

L

`links` (*BaseProduct* attribute), 13
`London` (*Region* attribute), 21

M

`Merseyside` (*Region* attribute), 21
`MeterPointDetails` (namedtuple in *octo_api.utils*), 19
 `gsp` (namedtuple field), 19
 `mpan` (namedtuple field), 19
 `profile_class` (namedtuple field), 19
`Midlands` (*Region* attribute), 21
`MIT License`, 24
module
 octo_api.api, 3
 octo_api.consumption, 8
 octo_api.pagination, 9
 octo_api.products, 11

octo_api.utils, 19
mpan (*namedtuple field*)
MeterPointDetails (*namedtuple in octo_api.utils*), 19

N

NightUnitRate (*RateType attribute*), 20
NorthEastern (*Region attribute*), 21
NorthScotland (*Region attribute*), 22
NorthWestern (*Region attribute*), 21

O

octo_api.api
module, 3
octo_api.consumption
module, 8
octo_api.pagination
module, 9
octo_api.products
module, 11
octo_api.utils
module, 19
OctoAPI (*class in octo_api.api*), 3
OctoResponse (*typeddict in octo_api.pagination*), 9

P

PaginatedResponse (*class in octo_api.pagination*), 9
Product (*class in octo_api.products*), 13
profile_class (*namedtuple field*)
MeterPointDetails (*namedtuple in octo_api.utils*), 19
Python Enhancement Proposals
PEP 517, 26

R

RateInfo (*class in octo_api.products*), 16
RegionalTariffs (*class in octo_api.products*), 19

S

sample_consumption (*DetailedProduct attribute*), 16
sample_quotes (*DetailedProduct attribute*), 16
single_register_electricity_tariffs
(*DetailedProduct attribute*), 16
single_register_gas_tariffs
(*DetailedProduct attribute*), 16
SouthEastern (*Region attribute*), 21
Southern (*Region attribute*), 21
SouthScotland (*Region attribute*), 22
SouthWales (*Region attribute*), 21
SouthWestern (*Region attribute*), 21
StandardUnitRate (*RateType attribute*), 20

StandingCharge (*RateType attribute*), 20

T

Tariff (*class in octo_api.products*), 18
tariffs_active_at (*DetailedProduct attribute*), 16
term (*BaseProduct attribute*), 13
to_dict() (*BaseProduct method*), 13
to_dict() (*Consumption method*), 9
to_dict() (*DetailedProduct method*), 16
to_dict() (*RateInfo method*), 17
to_dict() (*Tariff method*), 19

V

valid_from (*RateInfo attribute*), 17
valid_to (*RateInfo attribute*), 17
value_exc_vat (*RateInfo attribute*), 17
value_inc_vat (*RateInfo attribute*), 17

Y

Yorkshire (*Region attribute*), 21