AsyncDex

Release 0.2

PythonCoderAS

May 08, 2021

GETTING STARTED:

AsyncDex is an aiohttp-based async client for the MangaDex API. It respects all ratelimit rules set by MangaDex. Support for authentication is included as well as for running in anonymous mode.

CHAPTER

ONE

QUICKSTART

Warning: All AsyncDex operations have to be done inside of an async context.

Create an instance of MangadexClient:

```
from asyncdex import MangadexClient

async def main():
    client = MangadexClient()
```

Make a request for a random manga and print the authors of the manga:

```
manga = await client.random_manga()
print(f"{manga.id}: {manga.titles.en.primary}")
await manga.load_authors()
print(f"Author of {manga.titles.en.primary}: {manga.authors[0].name}")
```

Log in to your MangaDex account:

```
await client.login(username="yourusername", password="yourpassword")
# alternate method
client = MangadexClient(username="yourusername", password="yourpassword")
await client.login()
```

View your manga follows list:

```
response = await client.request("GET", "/user/follows/manga")
json = await response.json()
[print(item["data"]["id"]) for item in json["results"]]
```

CHAPTER

TWO

CHANGELOG

2.1 v0.2

2.1.1 Added

- **The 6 enums:**
    1. *Demographic*
    2. *MangaStatus*
    3. *FollowStatus*
    4. *ContentRating*
    5. *Visibility*
    6. *Relationship*
- *Missing*
- *InvalidID*
- **Models:**
    - *Model*
    - *Manga*
    - *Tag*
    - *Author*
- *tag_cache* inside of *MangadexClient*
- **Methods to *MangadexClient*:**
    - *refresh_tag_cache()*
    - *get_tag()*
    - *get_manga()*
    - *random_manga()*
    - *batch_authors()*
    - *get_author()*
    - *batch_mangas()*
- *DatetimeMixin*

- *TitleList*
- *AttrDict*
- *DefaultAttrDict*
- *copy_key_to_attribute()*
- *parse_relationships()*

2.2 v0.1

The initial release of AsyncDex.

CHAPTER

THREE

ASYNCDEX API

This page contains all of the classes, attributes, and methods of the various parts of AsyncDex.

3.1 Client

class asyncdex.**MangadexClient**(*, *username: Optional[str] = None*, *password: Optional[str] = None*, *refresh_token: Optional[str] = None*, *sleep_on_ratelimit: bool = True*, *session: Optional[aiohttp.client.ClientSession] = None*, *api_url: str = 'https://api.mangadex.org'*, *anonymous: bool = False*, ***session_kwargs*)

The main client that runs preforms all of the method requests.

> **Warning:** The client object should only be created under an async context. While it should be safe to initialize normally, the aiohttp ClientSession does not like this.

Parameters

- **username** (*str*) – The username of the user to authenticate as. Leave blank to not allow login to fetch a new refresh token. Specifying the username without specifying the password is an error.
- **password** (*str*) – The password of the user to authenticate as. Leave blank to not allow login to fetch a new refresh token. Specifying the password without specifying the username is an error.
- **refresh_token** (*str*) – The refresh token to use. Leaving the `username` and `password` parameters blank but specifying this parameter allows the client to make requests using the refresh token for as long as it is valid. Once the refresh token is invalid, if the username and password are not specified, the client will throw *Unauthorized*, unless *logout()* is used to set the client to anonymous mode.
- **sleep_on_ratelimit** (*bool*) – Whether or not to sleep when a ratelimit occurs or raise a *Ratelimit*. Defaults to True.
- **session** (*aiohttp.ClientSession*) – The session object for the client to use. If one is not provided, the client will create a new session instead. This is useful for providing a custom session.
- **anonymous** (*bool*) – Whether or not to force anonymous mode. This will clear the username and/or password.
- **session_kwargs** – Optional keyword arguments to pass on to the `aiohttp.ClientSession`.

anonymous_mode: *bool*
Whether or not the client is operating in **Anonymous Mode**, where it only accesses public endpoints.

api_base: *str*
The base URL for the MangaDex API, without a slash at the end.

async batch_authors(**authors: asyncdex.models.author.Author*)
Updates a lot of authors at once, reducing the time needed to update tens or hundreds of authors.

New in version 0.2.

Parameters **authors** (*Tuple[Author, ...]*) – A tuple of all the authors (and artists) to update.

async batch_mangas(**mangas: asyncdex.models.manga.Manga*)
Updates a lot of mangas at once, reducing the time needed to update tens or hundreds of mangas.

New in version 0.2.

Warning: If duplicate mangas are specified, only one manga instance will be updated.

Parameters **mangas** (*Tuple[Manga, ...]*) – A tuple of all the mangas to update.

get_author(*id: str*) → asyncdex.models.author.Author
Get an author using it's ID.

New in version 0.2.

Note: This method can also be used to get artists, since they are the same class.

Warning: This method returns a **lazy** Author instance. Call *Author.fetch()* on the returned object to see any values.

Parameters **id** (*str*) – The author's UUID.

Returns A *Author* object.

Return type *Author*

get_manga(*id: str*) → asyncdex.models.manga.Manga
Get a manga using it's ID.

New in version 0.2.

See also:

search().

Warning: This method returns a **lazy** Manga instance. Call *Manga.fetch()* on the returned object to see any values.

Parameters **id** (*str*) – The manga's UUID.

Returns A *Manga* object.

Return type *Manga*

async get_session_token()
Get the session token and store it inside the client.

async get_tag(*id: str*) → asyncdex.models.tag.Tag
Get a tag using it's ID.

New in version 0.2.

Note: In order to find a tag by the tag's name, it is recommended to call *refresh_tag_cache()* and then iterate over *tag_cache*.

Parameters **id** (*str*) – The tag's UUID.

Returns A *Tag* object.

Return type *Tag*

async login(*username: Optional[str] = None*, *password: Optional[str] = None*)
Logs in to the MangaDex API.

Parameters

- **username** (*str*) – Provide a username in order to make the client stop running in anonymous mode. Specifying the username without specifying the password is an error.
- **password** (*str*) – Provide a password in order to make the client stop running in anonymous mode. Specifying the password without specifying the username is an error.

async logout()
Log out from the API. If a refresh token exists, calls the logout route on the API. The username and password are cleared, and the client is put into anonymous mode.

password: Optional[str]
The password of the user that the client is logged in as. This will be None when the client is operating in anonymous mode.

async random_manga() → asyncdex.models.manga.Manga
Get a random manga.

New in version 0.2.

Returns A random manga.

Return type *Manga*

ratelimits: *asyncdex.ratelimit.Ratelimits*
The *Ratelimits* object that the client is using.

async refresh_tag_cache()
Refresh the internal tag cache.

New in version 0.2.

See also:

tag_cache

refresh_token: Optional[str]
The refresh token that the client has obtained. This will be None when the client is operating in anonymous mode, as well as if the client has not obtained a refresh token from the API.

async request(*method: str*, *url: str*, ***, *params: Optional[Mapping[str, Union[str, Sequence[str]]]] = None*, *json: Optional[Any] = None*, *with_auth: bool = True*, *retries: int = 3*, ***session_request_kwargs*) → aiohttp.client_reqrep.ClientResponse

Perform a request.

> **Warning:** All requests have to be released, otherwise connections will not be reused. Make sure to call `aiohttp.ClientResponse.release()` on the object returned by the method if you do not read data from the response.

Parameters

- **method** (*str*) – The HTTP method to use for the request.
- **url** (*str*) – The URL to use for the request. May be either an absolute URL or a URL relative to the base MangaDex API URL.
- **params** (*Mapping[str, Union[str, Sequence[str]]]*) – Optional query parameters to append to the URL. If one of the values of the parameters is an array, the elements will be automatically added to the URL in the order that the array elements appear in.
- **json** (*Any*) – JSON data to pass in a POST request.
- **with_auth** (*bool*) – Whether or not to append the session token to the request headers. Requests made without the header will behave as if the client is in anonymous mode. Defaults to `True`.
- **retries** (*int*) – The amount of times to retry. The function will recursively call itself, subtracting `1` from the original count until retries run out.
- **session_request_kwargs** – Optional keyword arguments to pass to `aiohttp.ClientSession.request()`.

Raises *Unauthorized* if the endpoint requires authentication and sufficient parameters for authentication were not provided to the client.

Returns The response.

Return type aiohttp.ClientResponse

session: aiohttp.client.ClientSession

The `aiohttp.ClientSession` that the client will use to make requests.

property session_token

The session token tht the client has obtained. This will be None when the client is operating in anonymous mode, as well as if the client has not obtained a refresh token from the API or if it has been roughly 15 minutes since the token was retrieved from the server.

sleep_on_ratelimit: bool

Whether or not to sleep when a ratelimit occurs.

tag_cache: Dict[str, asyncdex.models.tag.Tag]

A cache of tags. This cache will be used to lower the amount of tag objects, and allows for easily updating the attributes of tags. This cache can be refreshed manually by either calling *refresh_tag_cache()* or fetching data for any tag object.

New in version 0.2.

username: Optional[str]
The username of the user that the client is logged in as. This will be None when the client is operating in anonymous mode.

3.2 Models

class asyncdex.models.abc.**Model**(*client: MangadexClient*, ***, *id: Optional[str] = None*, *version: int = 0*, *data: Optional[Dict[str, Any]] = None*)
An abstract model. Cannot be instantiated.

New in version 0.2.

Parameters **data** (*Dict[str, Any]*) – The data received from the server. May be None if there is no data yet.

client: MangadexClient
The client that created this model.

abstract async fetch()
Fetch the data to complete any missing non-critical values. This method must call *parse()*.

id: str
A UUID that represents this item.

abstract parse(*data: Dict[str, Any]*)
Parse the data received from the server.

Parameters **data** (*Dict[str, Any]*) – The data from the server.

transfer(*new_obj: _T*)
Transfer data from a new object to the current object.

Parameters **new_obj** (*Model*) – The new object. Should be the same type as the current model.

version: int
The version of the model.

class asyncdex.models.**Manga**(*client: MangadexClient*, ***, *id: Optional[str] = None*, *version: int = 0*, *data: Optional[Dict[str, Any]] = None*)
A *Model* representing an individual manga.

New in version 0.2.

amazon_id: Optional[str]
The ID for the entry on Amazon, if it exists.

property amazon_url
Returns a formatted url for the manga's Amazon entry if it exists.

Note: While the MangaDex API currently returns fully formatted URLs for the *amazon_id* attribute, this may change in the future. This property will always return a fully formatted URL.

Returns A full URL or None if *amazon_id* is None.

Return type str

anilist_id: Optional[str]
The ID for the entry on Anilist, if it exists.

property anilist_url
Returns a formatted url for the manga's Anilist entry if it exists.

Returns A full URL or None if *anilist_id* is None.

Return type str

animeplanet_id: Optional[str]
The ID for the entry on AnimePlanet, if it exists.

property animeplanet_url
Returns a formatted url for the manga's AnimePlanet entry if it exists.

Returns A full URL or None if *animeplanet_id* is None.

Return type str

artists: List[asyncdex.models.author.Author]
A list of *Author* objects that represent the manga's artists.

See also:

authors

Note: In order to efficiently get all authors and artists in one go, use *load_authors()*.

---

authors: List[asyncdex.models.author.Author]
A list of *Author* objects that represent the manga's authors.

See also:

artists

Note: In order to efficiently get all authors and artists in one go, use *load_authors()*.

---

bookwalker_id: Optional[str]
The ID for the entry on Bookwalker, if it exists.

property bookwalker_url
Returns a formatted url for the manga's Bookwalker entry if it exists.

Returns A full URL or None if *bookwalker_id* is None.

Return type str

cdjapan_id: Optional[str]
The ID for the entry on CDJapan, if it exists.

property cdjapan_url
Returns a formatted url for the manga's CDJapan entry if it exists.

Note: While the MangaDex API currently returns fully formatted URLs for the *cdjapan_id* attribute, this may change in the future. This property will always return a fully formatted URL.

Returns A full URL or None if *cdjapan_id* is None.

Return type str

demographic: *asyncdex.enum.Demographic*
The manga's demographic.

descriptions: *asyncdex.utils.DefaultAttrDict*[Optional[str]]
A *DefaultAttrDict* holding the descriptions of the manga.

Note: If a language is missing a description, `None` will be returned.

ebookjapan_id: Optional[str]
The ID for the entry on EbookJapan, if it exists.

property ebookjapan_url
Returns a formatted url for the manga's EbookJapan entry if it exists.

Note: While the MangaDex API currently returns fully formatted URLs for the *ebookjapan_id* attribute, this may change in the future. This property will always return a fully formatted URL.

Returns A full URL or None if *ebookjapan_id* is None.

Return type str

english_translation_url: Optional[str]
The URL for the official English translation of the manga, if it exists.

async fetch()
Fetch the data to complete any missing non-critical values. This method must call *parse()*.

kitsu_id: Optional[str]
The ID for the entry on Kitsu, if it exists.

property kitsu_url
Returns a formatted url for the manga's Kitsu entry if it exists.

Returns A full URL or None if *kitsu_id* is None.

Return type str

last_chapter: Optional[int]
The last chapter of the manga. `None` if it is not specified or does not exist.

last_volume: Optional[int]
The last volume of the manga. `None` if it is not specified or does not exist.

async load_authors()
Shortcut method that calls `Client.batch_authors()` with the authors and artists that belong to the manga.

Roughly equivalent to:

```
await client.batch_authors(*manga.authors, *manga.artists)
```

locked: bool
A locked manga. Usually means that chapter details cannot be modified.

mangaupdates_id: Optional[str]
The ID for the entry on MangaUpdates, if it exists.

property mangaupdates_url
Returns a formatted url for the manga's MangaUpdates entry if it exists.

Returns A full URL or None if *mangaupdates_id* is None.

Return type str

myanimelist_id: Optional[str]
The ID for the entry on MyAnimeList, if it exists.

property myanimelist_url
Returns a formatted url for the manga's MyAnimeList entry if it exists.

Returns A full URL or None if *myanimelist_id* is None.

Return type str

novelupdates_id: Optional[str]
The ID for the entry on NovelUpdates, if it exists.

property novelupdates_url
Returns a formatted url for the manga's NovelUpdates entry if it exists.

Returns A full URL or None if *novelupdates_id* is None.

Return type str

original_language: str
The original language that the manga was released in.

parse(*data: Dict[str, Any]*)
Parse the data received from the server.

Parameters **data** (*Dict[str, Any]*) – The data from the server.

rating: *asyncdex.enum.ContentRating*
The manga's content rating.

raw_url: Optional[str]
The URL for the official raws of the manga, if it exists.

status: *asyncdex.enum.MangaStatus*
The manga's status.

tags: List[asyncdex.models.tag.Tag]
A list of *Tag* objects that represent the manga's tags. A manga without tags will have an empty list.

titles: *asyncdex.utils.DefaultAttrDict*[*asyncdex.models.title.TitleList*]
A *DefaultAttrDict* holding the titles of the manga.

year: Optional[int]
The year the manga started publication. May be None if publication hasn't started or is unknown.

class asyncdex.models.**Tag**(*client: MangadexClient*, ***, *id: Optional[str] = None*, *version: int = 0*, *data: Optional[Dict[str, Any]] = None*)
A *Model* representing a tag in a Manga.

New in version 0.2.

async fetch()
Fetch the data to complete any missing non-critical values. This method must call *parse()*.

names: *asyncdex.utils.DefaultAttrDict*[Optional[str]]
A *DefaultAttrDict* holding the names of the tag.

Note: If a language is missing a name, None will be returned.

---

parse(*data: Dict[str, Any]*)
Parse the data received from the server.

Parameters **data** (*Dict[str, Any]*) – The data from the server.

class asyncdex.models.**Author**(*client: MangadexClient*, *, *id: Optional[str] = None*, *version: int = 0*, *data: Optional[Dict[str, Any]] = None*)
A *Model* representing an individual author.

Note: Artists and authors are stored identically and share all properties.

New in version 0.2.

biographies: ***asyncdex.utils.DefaultAttrDict*[Optional[str]]**
A *DefaultAttrDict* holding the biographies of the author.

async fetch()
Fetch the data to complete any missing non-critical values. This method must call *parse()*.

image: **Optional[str]**
An image of the author, if available.

async load_mangas()
Shortcut method that calls Client.batch_mangas() with the mangas that belong to the author.

Roughly equivalent to:

```
await client.batch_mangas(*author.mangas)
```

mangas: **List[*Manga*]**
A list of all the mangas that the author has written.

Note: In order to efficiently get all mangas in one go, use:

```
await client.batch_mangas(*author.mangas)
```

name: **str**
The name of the author.

parse(*data: Dict[str, Any]*)
Parse the data received from the server.

Parameters **data** (*Dict[str, Any]*) – The data from the server.

3.3 Exceptions

exception asyncdex.exceptions.**AsyncDexException**
Base exception class for all exceptions by the package.

exception asyncdex.exceptions.**Ratelimit**(*path: str*, *ratelimit_amount: int*, *ratelimit_expires: datetime.datetime*)
An exception raised if MangadexClient.sleep_on_ratelimit is set to False.

path: **str**
The route that was taken that hit the ratelimit. This will match the path in the MangaDex API Documentation.

ratelimit_amount: int
How many calls to this path can be made once the ratelimit expires without being ratelimited again.

ratelimit_expires: datetime.datetime
A datetime.datetime object in UTC time representing when the ratelimit will expire.

exception asyncdex.exceptions.**HTTPException**(*path: str*, *response: aiohttp.client_reqrep.ClientResponse*)
Exceptions for HTTP status codes.

path: str
The URL taken that hit the error.

response: aiohttp.client_reqrep.ClientResponse
The aiohttp.ClientResponse object from the request.

exception asyncdex.exceptions.**Unauthorized**(*path: str*, *response: Optional[aiohttp.client_reqrep.ClientResponse]*)
An exception raised if a request to an endpoint requiring authorization is made where the client cannot authorize using provided information.

response: Optional[aiohttp.client_reqrep.ClientResponse]
The aiohttp.ClientResponse object from the request. May be None if a user tries to login without stored credentials.

exception asyncdex.exceptions.**Missing**(*attribute: str*, *model: Optional[str] = None*)
An exception raised if a response is missing a critical element for a model.

New in version 0.2.

Parameters **model** (*str*) – The name of the model that requires the attribute. Can be empty.

attribute: str
The name of the attribute that is missing.

exception asyncdex.exceptions.**InvalidID**(*id: str*, *model: Type[Model]*)
An exception raised if an invalid ID is given to any of the get_* methods representing that an item with this ID does not exist.

New in version 0.2.

id: str
The given ID

model: Type[Model]
The model that would have been returned had the ID been valid.

3.4 Enums

class asyncdex.enum.**Demographic**(*value*)
An Enum representing the various demographics. Source: https://api.mangadex.org/docs.html#section/Static-data/Manga-publication-demographic.

New in version 0.2.

JOSEI = 'josei'
A Josei Manga.

SEINEN = 'seinen'
A Seinen Manga.

SHOUJO = 'shoujo'
A Shoujo Manga.

SHOUNEN = 'shounen'
A Shounen Manga.

Note: In the developer documentation as of May 7, 2021, there is a typo in the word `Shounen`, where it is spelled without the `u`. However, the actual API will only recognize the variant including a `u`. For the library, both variations can be used for the enum.

---

class asyncdex.enum.**MangaStatus**(*value*)
An Enum representing the various statuses a manga can have. Source: https://api.mangadex.org/docs.html#section/Static-data/Manga-status

New in version 0.2.

Note: The status of the manga does not dictate whether or not the chapter list will be stable. Scanlation teams may have not published all chapters up to the completion of updates, so the manga may still get new chapters, especially in different languages. The only way to know if a manga has actually finished updating is by checking if the "end chapter" is present in the current language. Even this is not a guarantee, as an author may add additional media accompanying the work, such as a extra page related to the manga on Twitter or Pixiv, especially for manga that are mainly published online. The labels shown for a manga's status should not dictate the policy for update checking, as they are only meant to be an aid for end users and not actually representative of the immutability of the manga's chapter list.

ABANDONED = 'abandoned'
A manga where the author has intentionally stopped publishing new chapters.

COMPLETED = 'completed'
A manga that has finished publication.

HIATUS = 'hiatus'
A manga where the author is on a known hiatus.

ONGOING = 'ongoing'
A manga that is actively being published, in volume format, in a magazine like Weekly Shonen, or online.

class asyncdex.enum.**FollowStatus**(*value*)
An Enum representing the status that the user has marked the manga has. Source: https://api.mangadex.org/docs.html#section/Static-data/Manga-reading-status

New in version 0.2.

COMPLETED = 'completed'
A manga that the user has marked as completed.

> **Warning:** When a manga is marked as completed, the MangaDex API drops all chapter read markers. Setting a manga as completed **will** result in the deletion of data. Be very careful!

DROPPED = 'dropped'
A manga that the user has marked as dropped.

ON_HOLD = 'on_hold'
A manga that the user has marked as on_hold.

PLAN_TO_READ = 'plan_to_read'
A manga that the user has marked as plan_to_read.

READING = 'reading'
A manga that the user has marked as reading.

RE_READING = 're_reading'
A manga that the user has marked as re_reading.

class asyncdex.enum.**ContentRating**(*value*)
An Enum representing the content in a manga. Source: https://api.mangadex.org/docs.html#section/Static-data/Manga-content-rating

New in version 0.2.

EROTICA = 'erotica'
A manga that is erotica.

Note: This type of content represents content tagged with the `Smut` tag.

PORNOGRAPHIC = 'pornographic'
A manga that is pornographic.

Note: This type of content was the only type of content that MangaDex's old 18+ filter used to block. This type of content was also the type of content that old MangaDex APIs used to call "hentai".

SAFE = 'safe'
A manga that is safe.

Note: This is the only content rating that means a manga is safe for work. All other values are not safe for work (NSFW).

SUGGESTIVE = 'suggestive'
A manga that is suggestive.

Note: This type of content represents content tagged with the `Ecchi` tag.

class asyncdex.enum.**Visibility**(*value*)
An enum representing the visibility of an `CustomList`. Source: https://api.mangadex.org/docs.html#section/Static-data/CustomList-visibility

New in version 0.2.

PRIVATE = 'private'
A private `CustomList`.

PUBLIC = 'public'
A public `CustomList`.

class asyncdex.enum.**Relationship**(*value*)
An enum representing the different types of relationship types. Source: https://api.mangadex.org/docs.html#section/Static-data/Relationship-types

New in version 0.2.

ARTIST = 'artist'
A *Author* resource.

AUTHOR = 'author'
A *Author* resource.

CHAPTER = 'chapter'
A Chapter resource.

CUSTOM_LIST = 'custom_list'
A CustomList resource.

MANGA = 'manga'
A *Manga* resource.

SCANLATION_GROUP = 'scanlation_group'
A Group resource.

TAG = 'tag'
A *Tag* resource.

USER = 'user'
A User resource.

3.5 Constants

asyncdex.constants.**ratelimit_data: List[*asyncdex.ratelimit.PathRatelimit*]**
These are the ratelimit rules taken from the API Docs.

Note: The API rules given here do not reflect all possible API ratelimit rules. The client will automatically ratelimit when appropriate headers are sent by the API. Check the latest API rules at the official API documentation.

New in version 0.1.

asyncdex.constants.**routes: Dict[str, str]**
The various predefined routes for the client. If the API changes for a given destination, the route can easily be modified without copy-pasting the route to the functions using it.

3.6 Ratelimit

class asyncdex.ratelimit.**Path**(*name: str*, *path_regex: re.Pattern*, *method: Optional[str] = None*)
A Path object representing a various path.

method: Optional[str] = None
The HTTP method for the path. Leave None if ratelimit applies to all methods.

name: str
The name of the path. This will be the value provided by *Ratelimit.path*.

path_regex: re.Pattern
A compiled regex pattern matching the path, used when the path has a variable, such as /action/{id}.

class asyncdex.ratelimit.**PathRatelimit**(*path:* asyncdex.ratelimit.Path, *ratelimit_amount: int*, *ratelimit_time: int*)
An object that allows the request method to check the ratelimit before making a response.

can_call(*method: str*) → bool
Returns whether or not this route can be used right now.

Parameters **method** (*str*) – The HTTP method being used.

Returns Whether or not this route can be used without ratelimit.

Return type bool

expire()
Expire the ratelimit.

path: ***asyncdex.ratelimit.Path***
A *Path* object.

ratelimit_amount: **int**
Analogous to *Ratelimit.ratelimit_amount*

ratelimit_expires: **datetime.datetime = datetime.datetime(1, 1, 1, 0, 0)**
Analogous to *Ratelimit.ratelimit_expires*

ratelimit_time: **int**
The amount of time needed for the ratelimit to expire after the first use.

ratelimit_used: **int = 0**
How many times the path has been called since the last ratelimit expire.

time_until_expire() → datetime.timedelta
Returns a *datetime.timedelta* representing the amount of seconds for the ratelimit to expire.

update(*response: aiohttp.client_reqrep.ClientResponse*)
Update the path's ratelimit based on the headers.

Parameters **response** (*aiohttp.ClientResponse*) – The response object.

class asyncdex.ratelimit.**Ratelimits**(**ratelimits:* asyncdex.ratelimit.PathRatelimit)
An object holding all of the various ratelimits.

Parameters **ratelimits** (*PathRatelimit*) – The *PathRatelimit* object.

add(*obj:* asyncdex.ratelimit.PathRatelimit)
Add a new ratelimit. If the path is the same as an existing path, it will be overwritten.

Parameters **obj** (*PathRatelimit*) – The new ratelimit object to add.

async check(*url: str*, *method: str*) → Tuple[float, Optional[*asyncdex.ratelimit.PathRatelimit*]]
Check if a path is ratelimited.

Parameters

- **url** (*str*) – The path, starting with /
- **method** (*str*) – The HTTP method being used.

Returns A number representing the amount of seconds before ratelimit expire or -1 if there is no need to ratelimit as well as the *PathRatelimit* object if found.

Return type float

ratelimit_dictionary: **Dict[re.Pattern,** ***asyncdex.ratelimit.PathRatelimit*****]**
A dictionary where the keys are regex patterns representing the paths and the values are *PathRatelimit* objects.

remove(*obj:* asyncdex.ratelimit.PathRatelimit)
Remove a ratelimit.

Parameters **obj** (PathRatelimit) – The new ratelimit object to remove.

async sleep(*url: str*, *method: str*) → Optional[*asyncdex.ratelimit.PathRatelimit*]
Helper function that sleeps the amount of time returned by *check()*.

Parameters

- **url** (*str*) – The path, starting with /
- **method** (*str*) – The HTTP method being used.

Returns The *PathRatelimit* object if found

Return type *PathRatelimit*

3.7 Misc

asyncdex.utils.**remove_prefix**(*prefix: str*, *string: str*) → str
Remove a prefix from a string. This is a polyfill for Python versions <3.9.

Parameters

- **prefix** (*str*) – The prefix to remove
- **string** (*str*) – The string to remove the prefix from

Returns The string without the prefix

Return type str

class asyncdex.utils.**AttrDict**
A dict where keys can be accessed by attributes.

New in version 0.2.

class asyncdex.utils.**DefaultAttrDict**(*mapping_or_iterable: Optional[Union[Mapping[str, _VT], Iterable[Tuple[str, _VT]]]] = None*, ***, *default: Callable[], _VT]*)
A *AttrDict* with a default.

New in version 0.2.

default
A callable that accepts no arguments and returns an instance of the value's class.

asyncdex.utils.**copy_key_to_attribute**(*source_dict: dict*, *key: str*, *obj: Any*, *attribute_name: Optional[str] = None*, ***, *default: Any = Sentinel*, *transformation: Optional[Callable[[str], Any]] = None*)
Copies the value of a dictionary's key to an object.

New in version 0.2.

Parameters

- **source_dict** (*dict*) – The dictionary with the key and value.
- **key** (*str*) – The key that has the value.
- **obj** (*Any*) – The object to set the attribute of.
- **attribute_name** (*str*) – The name of the attribute to set if the name of the key and the name of the attribute are different.
- **default** (*Any*) – A default value to add if the value is not found.

- **transformation** (*Callable[[str], Any]*) – A callable that will be executed on the value of the key. It should accept a `str` and can return anything.

`asyncdex.utils.`**parse_relationships**(*data: dict*, *obj: Model*)

Parse the relationships available in a model.

New in version 0.2.

Parameters

- **data** (*dict*) – The raw data received from the API.
- **obj** (`Model`) – The object to add the models to.

class `asyncdex.models.mixins.`**DatetimeMixin**

A mixin for models with `created_at` and `updated_at` attributes.

New in version 0.2.

created_at: datetime.datetime

A `datetime.datetime` representing the object's creation time.

See also:

modified_at()

Note: The datetime is **timezone aware** as it is parsed from an ISO-8601 string.

property modified_at

The last time the object was modified. This will return the creation time if the object was never updated after creation, or the modification time if it has.

See also:

created_at

See also:

updated_at

Returns

The last time the object was changed as a `datetime.datetime` object.

Note: The datetime is **timezone aware** as it is parsed from an ISO-8601 string.

Return type `datetime.datetime`

updated_at: Optional[datetime.datetime]

A `datetime.datetime` representing the last time the object was updated. May be None if the object was never updated after creation.

See also:

modified_at()

Note: The datetime is **timezone aware** as it is parsed from an ISO-8601 string.

class asyncdex.models.title.**TitleList**(*iterable=()*, */*)

An object representing a list of titles.

New in version 0.2.

property primary

Returns the primary title for the language if it exists or else returns None.

Returns The first title in the list.

Return type str

INDEX

S

T

U

V

Y