RAWGpy Documentation

Release 1.0

Laurin Schmidt

Dec 07, 2022

Contents

These API docs use terminology from the RAWG API. Important to note may be:

slug The slug is the version of a name that was adjusted for use in URLs, spaces are replaced with hyphens and all characters are lowercase

CHAPTER 1

RAWGpy quickstart

The RAWGpy RAWG.io API wrapper uses the *rawgpy.rawg.RAWG* as a main class that the users accesses.

You can use the RAWGpy wrapper with or without authenticating.

```
import rawgpy

rawg = rawgpy.RAWG("User-Agent, this should identify your app")
results = rawg.search("Warframe")  # defaults to returning the top 5 results
game = results[0]
game.populate()  # get additional info for the game

print(game.name)

print(game.description)

for store in game.stores:
    print(store.url)

rawg.login("someemail@example.com", "somepassword")

me = rawg.current_user()

print(me.name) # print my name, equivalent to print(self.username)

me.populate() # gets additional info for the user

for game in me.playing:
    print(game.name) # prints all the games im currently playing
```

CHAPTER 2

RAWGpy's Data Classes

rawgpy.data_classes.charts	Classes that represent a games chart listing values on rawg
rawgpy.data_classes.id_name_slug	Classes that mostly only have the attributes *id*, *name* and *slug*
*rawgpy.data_classes.platform_*	Represents a Platform that a game is availiable on
rawgpy.data_classes.rating	Represents a Rating of a game
rawgpy.data_classes.store	Represents a Store that a game is availiable on

2.1 charts

Classes that represent a games chart listing values on rawg

class rawgpy.data_classes.charts.**Chart**(*position*, *change*)
parent Chart class

class rawgpy.data_classes.charts.**GenreChart**(*genre*, *position*, *change*)
This represents the genre based chart list entry.

if a Game were to have a GenreChart with:

- :attr:position = 2
- :attr:genre = "Shooter"

it would be the second best/most popular shooter game on RAWG

class rawgpy.data_classes.charts.**YearChart**(*year*, *position*, *change*)
a YearChart

if a Game were to have a YearChart with:

- :attr:position = 2
- :attr:year = 2012

it would be the second best/most popular game on rawg in 2012

2.2 id_name_slug

Classes that mostly only have the attributes *id*, *name* and *slug*

class rawgpy.data_classes.id_name_slug.**Category**(*id_*, *name*, *slug*, *games_count*)

class rawgpy.data_classes.id_name_slug.**Developer**(*id_*, *name*, *slug*, *games_count*)

class rawgpy.data_classes.id_name_slug.**ESRB**(*id_*, *name*, *slug*)

class rawgpy.data_classes.id_name_slug.**Genre**(*id_*, *name*, *slug*, *games_count*)

class rawgpy.data_classes.id_name_slug.**IdNameSlug**(*id_*, *name*, *slug*)
base id, name, slug class that is jsut used as a parent to many others

class rawgpy.data_classes.id_name_slug.**Publisher**(*id_*, *name*, *slug*, *games_count*)

class rawgpy.data_classes.id_name_slug.**SimplePlatform**(*id_*, *name*, *slug*)

class rawgpy.data_classes.id_name_slug.**SimpleStore**(*id_*, *name*, *slug*)

class rawgpy.data_classes.id_name_slug.**Tag**(*id_*, *name*, *slug*, *games_count*)

2.3 platform_

Represents a Platform that a game is availiable on

class rawgpy.data_classes.platform_.**Platform**(*id_*, *name*, *slug*, *image*, *year_end*, *year_start*, *games_count*, *released_at*, *minimum_requirements*, *maximum_requirements*)

Represents a Platform

games_count = None
the number of games on this Platform

id = None
the id of the Platform

image = None
the image url for the Platform

maximum_requirements = None
the optimnal requirements for the game

minimum_requirements = None
the minimum requirements for the game

name = None
the name of the Platform

released_at = None
the time this Platform was released at

slug = None
the slug of the Platform

year_end = None
TODO

year_start = None
TODO

2.4 rating

Represents a Rating of a game

class rawgpy.data_classes.rating.**Rating**(*id_*, *title*, *count*, *percent*)
Represents a rating

count = None
the amount of these ratings

id = None
the ID of the rating

percent = None
the percent this rating occupies

title = None
the title of the rating

2.5 store

Represents a Store that a game is availiable on

rawgpy.rawg	The main rawg class
rawgpy.game	The class representing a game
rawgpy.base	The base class for converting from JSON
rawgpy.collection	The class representing a collection
rawgpy.user	The class representing a User
rawgpy.utils	Utility functions and classes

CHAPTER 3

rawg

The main rawg class

class rawgpy.rawg.**RAWG**(*user_agent*)

main RAWG class

the main class used for interactions with the RAWG.io database

collection_games(*slug*)

generator that yields the collections games json

Parameters **slug** (*str*) – the slug of the collection

Returns generator of the games json

Return type pagination_generator

collection_request(*slug*) → dict

Returns the collection json

Parameters **slug** (*str*) – the collection slug

Returns The collection json

Return type dict

create_collection(*name*, *description*) → rawgpy.collection.Collection

creates a new collection

Parameters

- **name** (*str*) – Name of the collection
- **description** (*str*) – Description of the collection

Returns Collection object of created collection

Return type *Collection*

current_user() → rawgpy.user.User

Returns the currently authenticated user

Returns The currently authenticated user

Return type *User*

current_user_request() → dict
Returns the currently authenticated user json

Returns Json of the authenticated user

Return type dict

game_collections(*game_slug*)
Retrieve the collections a game is a part of

Parameters **game_slug** (*str.*) – the slug of the game

Returns generator of the collections json

Return type pagination_generator

game_request(*slug*, *additional_param=''*) → dict
uses the `get_request` method to get a specific games json

Parameters

- **slug** (*str*) – the slug of the game that shoudl be returned, needs to be correct rawg slug
- **additional_param** – any additional request parameter

Returns json-like list / dict structure of the returned json

Return type dict

game_suggestions(*slug*)
generator that yields the suggested games for a game

Parameters **slug** (*str*) – the game slug

Returns generator of the games json

Return type pagination_generator

get_collection(*slug*) → rawgpy.collection.Collection
Returns the collection object

Parameters **slug** (*str*) – the collection slug

Returns The collection object

Return type *Collection*

get_game(*slug*) → rawgpy.game.Game
get a specific game

Parameters **slug** (*str*) – the slug of the game

Returns the game object

Return type *Game*

get_request(*url*) → dict
Sends a GET request

Parameters **url** (*str*) – the url it sends a request to,

Returns json-like list / dict structure of the returned json

Return type dict

get_user(*slug*) → rawgpy.user.User
gets a user *User*

Parameters **slug** (*str*) – the userslug

Returns user json

Return type dict

Returns generator of the collections json

Return type *User*

login(*email*, *password*)
Logs the user in, autheticating all subsequent requests with that user

Parameters

- **email** (*str*) – The users email
- **password** (*str*) – The users password

pagination_generator(*url*, *results_name='results'*, *next_name='next'*) → dict
Generator for pagination based urls, reads the value of *next_name* from the json to get the url for the next request

uses `get_request`

Parameters

- **url** (*str*) – the url, needs to return some kind of next url value
- **results_name** (*str, optional*) – the json key used to return the result, defaults to "results"
- **next_name** (*str, optional*) – the key used to get the next url, defaults to "next"

Returns the paginated objects json

Return type dict

patch_game(*slug*, *data*)
Patch (edit) a game

Parameters

- **slug** (*str*) – The games slug
- **data** (*dict*) – The edited data

Returns The response json

Return type dict

patch_request(*url*, *data*) → dict
Sends a PATCH request

Parameters

- **url** (*str*) – The PATCH url
- **data** (*dict*) – The PATCH data

Returns The response json

Return type dict

post_request(*url*, *data*) → dict
Sends a POST request

Parameters

- **url** (*str*) – The POST url
- **data** (*dict*) – The POST data

review_data(*level*, *reactions=None*, *add_to_library=False*, *text=''*, *post_twitter=False*, *post_facebook=False*) → dict
creates the data for review operations

Parameters

- **game_id** (*int*) – the int id of the game
- **level** (*int*) – 1 = skip, 3 = meh, 4 = Recommended, 5 = Exceptional
- **reactions** (*List[int], optional*) – List of the reactions to add, defaults to None
- **add_to_library** (*bool, optional*) – whether the game whouls be added to the authenticated users library, defaults to False
- **text** (*str, optional*) – the review text, uses html formats like
, defaults to ""
- **post_twitter** (*bool, optional*) – whether to post on twitter, defaults to False
- **post_facebook** (*bool, optional*) – whether to post on facebook, defaults to False

Returns The data that has been made

Return type dict

review_game(*game_id: int*, *level*, *reactions=None*, *add_to_library=False*, *text=''*, *post_twitter=False*, *post_facebook=False*)
Adds a review to a game

Parameters

- **game_id** (*int*) – the int id of the game
- **level** (*int*) – 1 = skip, 3 = meh, 4 = Recommended, 5 = Exceptional
- **reactions** (*List[int], optional*) – List of the reactions to add, defaults to None
- **add_to_library** (*bool, optional*) – whether the game whouls be added to the authenticated users library, defaults to False
- **text** (*str, optional*) – the review text, uses html formats like
, defaults to ""
- **post_twitter** (*bool, optional*) – whether to post on twitter, defaults to False
- **post_facebook** (*bool, optional*) – whether to post on facebook, defaults to False

search(*query*, *num_results=5*, *additional_param=''*) → rawgpy.game.Game
searches for games

Parameters

- **query** (*str*) – the search query
- **num_results** (*int, optional*) – the amount of results, defaults to 5
- **additional_param** (*str, optional*) – additional get parameters, defaults to ""

search_request(*query*, *num_results=5*, *additional_param=''*) → dict
uses the `get_request` method to search for a game

Parameters

- **query** (*str*) – the name of the game that should be searched for
- **num_results** (*int*) – the amount of results the search should return
- **additional_param** – any additional search parameter, like *&sorting=-_score* to sort the games by relevance, excluding popularity

Returns json-like list / dict structure of the returned json

Return type dict

user_games(*slug*, *status=None*)
generator that yields the users games json

Parameters

- **slug** (*str*) – the users slug
- **status** (*str, optional*) – the status of the game in the users library, can be *playing*, *owned*, *beaten*, *dropped*, *toplay*, *yet* defaults to "playing"

Returns generator of the games json

Return type pagination_generator

user_request(*slug*) → dict
Returns a user json

Parameters **slug** (*str*) – the users slug

Returns the users json

Return type dict

CHAPTER 4

game

The class representing a game

class rawgpy.game.**Game**(*json*)

The class representing a Game

categories

The categories that apply to this game, list of instances of *Category*

charts

The charts for this game, simply a tuple of _genrechart and _yearchart

The GenreChart shows what genre this game is most popular in The YearChart shows the popularity of the game in its release year

collect(*collection: rawgpy.collection.Collection*)

Adds this game to the provided collection

Parameters **collection** (*Collection*) – the collection object this game should be added to

collections

Returns a list of unpopulated Collection objects that this game is part of.

developers

The developers that worked on this game, list of instances of *Developer*

edit()

Sends all edited base variables to rawg

esrb

The ESRB rating this game got, of class ESRB

genres

The genres this game falls under, of class *Genre*

platforms

The platforms the game is availiable on, list of instances of *Platform*

populate()
Populates the game by re-requesting the data

publishers
The publishers this game was published by, of class *Publisher*

reactions
The reactions to the game, a list of dictionaries with reaction id as key and amount as value

review(*text: str*, *level: str*, *reaction=None*)
Adds a review to the game, only works if user is authenticated

Parameters

- **text** (*str*) – the review text, empty for none
- **level** (*str*) – the name of the rating, as shown on the website
- **reaction** (*List[int], optional*) – a list of reactions, defaults to None

stores
The stores the game is avaliable on, list of instances of `Store`

suggestions
Generator that returns *Game* instances of suggestions made by the rawg Neural Network modell

tags
The tags this game is tagged with, of class *Tag*

CHAPTER 5

base

The base class for converting from JSON

class rawgpy.base.**FromJSONobject**(*json*)
a base object that provides functionality for converting from json

CHAPTER 6

collection

The class representing a collection

class rawgpy.collection.**Collection**(*json*)

Class representing a collection of Games

add(*game*)

Adds a game to the collection, requires you to be owner of it

Parameters **game** (*Game*, or a list of.) – the game to be adde

creator

Returns the unpopulated *User* that made this collection

games

Returns a list of unpopulated *Game* objects that were added to this collection.

is_mine

Returns true if the collection is created by the currently authenticated user, requires authentication

populate()

Populates the collection by re-requesting the data

CHAPTER 7

user

The class representing a User

class rawgpy.user.**User**(*json*)

beaten
The games this user has makers as beaten, list of *Game*

dropped
The games this user has makers as dropped, list of *Game*

games
Returns a list of unpopulated *Game* objects that this user has added.

owned
The games this user has makerd as owned, list of *Game*

playing
The games this user has makers as playing, list of *Game*

populate()
Populates the user by re-requesting the data

toplay
The games this user has makers as toplay, list of *Game*

yet
The games this user has makers as yet, list of *Game*

CHAPTER 8

utils

Utility functions and classes

rawgpy.utils.**del_none**(*obj*)

recursive function for deleting none from json-dict

recursively removes none from a nested python data structure (dict, tuple, list, thing)

Parameters **obj** (*dict, list, set, tuple*) – the object none values are to be removed from

Returns the object without none values

Return type dict, list, set, tuple

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

Python Module Index

r

Index